Учреждение образования
«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ
ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

А. С. Кобайло

СИНТЕЗ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ РЕАЛЬНОГО ВРЕМЕНИ

ЛАБОРАТОРНЫЙ ПРАКТИКУМ

Рекомендовано
учебно-методическим объединением учреждений высшего образования Республики Беларусь по образованию в области информатики и радиоэлектроники в качестве пособия по дисциплине «Проектирование информационных систем» для студентов учреждений высшего образования по направлению специальности 1-40 01 02-03 «Информационные системы и технологии (издательско-полиграфический комплекс)»

Минск 2012

УДК 004.031.43(076.5)
ББК 32.973.36-018.1я73
К55

Рецензенты:
кафедра прикладной математики и информатики Института непрерывного образования БГУ (заведующий кафедрой кандидат технических наук, доцент *В. Д. Дубовец*);
доцент кафедры информационных технологий автоматизированных систем Белорусского государственного университета информатики и радиоэлектроники кандидат технических наук, доцент *О. В. Герман*

Кобайло, А. С.

К55 Синтез вычислительных систем реального времени. Лабораторный практикум : пособие по дисциплине «Проектирование информационных систем» для студентов по направлению специальности 1-40 01 02-03 «Информационные системы и технологии (издательско-полиграфический комплекс)» / А. С. Кобайло. – Минск : БГТУ, 2012. – 97 с.

ISBN 978-985-530-179-1.

Лабораторный практикум содержит основные понятия, теоретические сведения, методические указания, упражнения и задания для проведения лабораторных работ по курсу «Проектирование информационных систем». Пособие предназначено для студентов специальности 1-40 01 02-03 «Информационные системы и технологии (издательско-полиграфический комплекс)», а также может быть полезно тем студентам, которые занимаются проблемами разработки специального программного обеспечения и его реализации.

УДК 004.031.43(076.5)
ББК 32.973.36-018.1я73

ISBN 978-985-530-179-1

ВВЕДЕНИЕ

В соответствии с современной классификацией информационных систем (ИС) классификационный признак **сфера применения** в качестве одного из важнейших классов ИС выделяет ИС автоматизированного проектирования (САПР). ИС автоматизированного проектирования предназначены для автоматизации функций инженеров-проектировщиков, конструкторов, архитекторов, дизайнеров при создании новой техники или технологии. Основными функциями подобных систем являются: инженерные расчеты, создание графической документации (чертежей, схем, планов), создание проектной документации, моделирование проектируемых объектов, синтез схем.

В качестве предметной области автоматизации проектирования выберем вычислительные системы (ВС) с нетрадиционной архитектурой, в частности системы реального времени. Под ВС реального времени (ВСРВ) будем понимать ВС, работающие в режиме реального времени – режиме обработки данных, при котором взаимодействие ВС с внешними по отношению к ним процессами осуществляется в моменты, определяемые скоростью протекания этих процессов.

Анализ состояния в области автоматизации проектирования сложных технических систем показывает, что известные САПР, применение которых возможно при решении задач автоматизации проектирования средств вычислительной техники, предназначены, в первую очередь, для решения инженерно-конструкторских и технологических задач, таких как проектирование принципиальных схем по готовым функциональным схемам (для сложных задач используются САПР OrCAD (PSpiceA/D) и SPECCTRA, P-CAD 2000–200X (ACCELEDA) и AltiumDesigner (Protel), eProductDesigner, PowerPCB, CAM 350, Viewlogik (Analog), BETASoft, MATLAB+Simulink и т. д.), проектирование печатных плат (программа PeakFPGA компании Altium, модуль PLD, входящий в состав пакета Protel компании Altium, программа FPGAStudio компании Cadence Design Systems, программы Fusion/SpeedWave, Fusion/VSCi, Fusion/ViewSim, ViewPLD компании Innoveda, пакет программ SystemView компании Elanix), анализ электромагнитной совместимости (SpeedXP), проектирования ПЛИС (интегрированный пакет

Microwave Office 200X компании AWR, система полного электромагнитного моделирования EMPIRE компании IMST, система полного электромагнитного моделирования QuickWave-3D компании QWED, система полного электромагнитного моделирования CST Microwave Studio компании CST), электронных схем и чертежей (Модуль Elektra-CAD компании DesktopEDA для пакета Protel, пакет WSCAD компании WSCAD Electronic, пакет PCschematic Elautomation компании DpS CAD-center ApS, пакет Autocad Electrical компании Autodesk), конструкции устройства (системы AutoCAD, ProEngineer и SolidWorks, программа CADSTAR фирмы Zuken), а также для моделирования электронных схем на поведенческом уровне (пакет SystemView компании Elanix; пакет Microwave Office компании AWR). В то же время отсутствуют системы для автоматизации наиболее интеллектуальных этапов проектирования – структурного и функционального синтеза. Последнее обстоятельство обусловливает актуальность проблемы разработки программных средств автоматизации функционального проектирования вычислительных систем с нетрадиционной архитектурой.

Теоретической базой для создания программного продукта будут являться положения теории синтеза вычислительных систем реального времени (ТСВСРВ).

Лабораторная работа № 1
ФОРМИРОВАНИЕ ГРАФА ВЫЧИСЛИТЕЛЬНОГО АЛГОРИТМА

Целью лабораторной работы является изучение методов машинного представления графа вычислительного алгоритма реализации математической модели проектируемой системы.

1.1. Постановка задачи синтеза параллельно-конвейерных архитектур

1.1.1. Общие требования реального времени. Вычислительная система реального времени реализует вычислительный процесс общего вида

$$Y = F(X, Z, C), \tag{1.1}$$

где Y, X, Z – множества выходных, входных, промежуточных переменных; C – множество констант; при этом выполняется хотя бы одно из следующих требований.

1. Скорости обработки переменных $\varphi_i = X \cup Y \cup Z$, $i = \overline{1, I}$ должны соответствовать заданным соотношениям

$$\tau_{\varphi_i} \le \Delta t(\varphi_i), \tag{1.2}$$

где I – количество вершин графа алгоритма реализации процесса (1.1); τ_{φ_i} – время выполнения операции формирования переменной φ_i; $\Delta t(\varphi_i)$ – фиксированный интервал времени (шаг дискретизации), который задает периодичность обработки данных для получения значения переменной φ_i, в частности, для входных данных это период поступления на соответствующий вход вычислительной структуры очередного дискретного отсчета переменной φ_i; при этом справедливо:

$$\forall j, \subset \{i\}: \Delta t\left(\varphi_j\right) = \Delta t(\varphi_i)_{\min} \cdot k_n,\ k_n = \overline{1, K},$$
$$\Delta t(\varphi_i)_{\min} = \min_i \Delta t(\varphi_i),$$

где K – мощность множества интервалов $\Delta t(\varphi_n)$, $n \in \{i\}$:

$$r, s \subset \{n\}: \Delta t(\varphi_r) \neq \Delta t(\varphi_s).$$

2. При известных моментах $\{t(x_n)\}$, $x_n \in X$ поступления на входы структуры ряда переменных $x = \{x_n\}$ заданы моменты $\{t'(y_m)\}$ потребности в ряде выходных переменных $y = \{y_m\}$, $y_m \in Y$. Это требование можно определить в виде соотношения

$$t_{\text{ф}}(y_m) - t(x_n) \leq \Delta t(y_m, x_n), \tag{1.3}$$

где $t_{\text{ф}}(y_m)$ – момент завершения формирования переменной y_m; $\Delta t(y_m, x_n)$ – разностный интервал, по завершению которого после подачи на соответствующий вход переменной x_n должно быть получено значение y_m:

$$\Delta t(y_m, x_n) = t'(y_m) - t(x_n).$$

3. Время реализации алгоритма вычислительной структурой $T_{\text{а}}$ не должно превышать заданного значения:

$$T_{\text{а}} \leq T_{\text{зад}}. \tag{1.4}$$

При случайном характере временных параметров вычислительного процесса в правой части выражений (1.2)–(1.4) используется значение нижней границы существования соответствующего переменного параметра времени

$$\Delta T = \begin{cases} \Delta t_{\min}(\varphi_i), \\ \Delta T_{\min}(y_m, x_n), \\ T_{\text{зад min}}. \end{cases} \tag{1.5}$$

1.1.2. Представление модели вычислительного процесса в виде графа. Модель вычислительного процесса (1.1) определяется в виде графа алгоритма $G = (V, E)$, множество вершин V которого соответствует множеству операций отдельных фрагментов алгоритма, множество E ребер – информационным связям. Например, математической модели

$$Y = f_1(f_2(x_1 + x_2, x_3)) + (x_3 \cdot x_4, f_1(x_5 + x_6)) \tag{1.6}$$

соответствует граф, изображенный на рис. 1.1. Спецификация его вершин представлена в табл. 1.1.

Таблица 1.1

Спецификация вершин графа вычислительного алгоритма реализации математической модели формирования *y*

v_i	v_1	v_2	v_3	v_4	v_5	v_6	v_7
φ_i	Ввод x_1	Ввод x_2	Ввод x_3	Ввод x_4	Ввод x_5	Ввод x_6	+

Окончание табл. 1.1

v_i	v_8	v_9	v_{10}	v_{11}	v_{12}	v_{13}	v_{14}	v_{15}
φ_i	*	+	F_2	f_1	f_1	f_2	+	Вывод Y

Для машинного представления топологии графа при автоматизации синтеза ВС могут быть использованы известные способы его описания. Оценку эффективности того или иного способа представления графа будем производить с помощью двух параметров – объема памяти Q_M, требуемого для хранения информации о топологии графа, и коэффициента использования памяти K_J.

Коэффициентом использования памяти будем называть отношение количества информативных (ненулевых) ячеек к общему объему памяти, используемой для хранения массива информации:

$$K_J = Q_{inf} / Q_M.$$

Граф $G = (V, E)$ может быть описан матрицей инцидентностей, которая имеет I строк, J столбцов и элементы a_{ij}:

$$a_{ij} = \begin{cases} 1, \text{ если } j\text{-я дуга выходит из } i\text{-й вершины,} \\ -1, \text{ если } j\text{-я дуга входит в } i\text{-ю вершину,} \\ 0 \text{ в остальных случаях.} \end{cases}$$

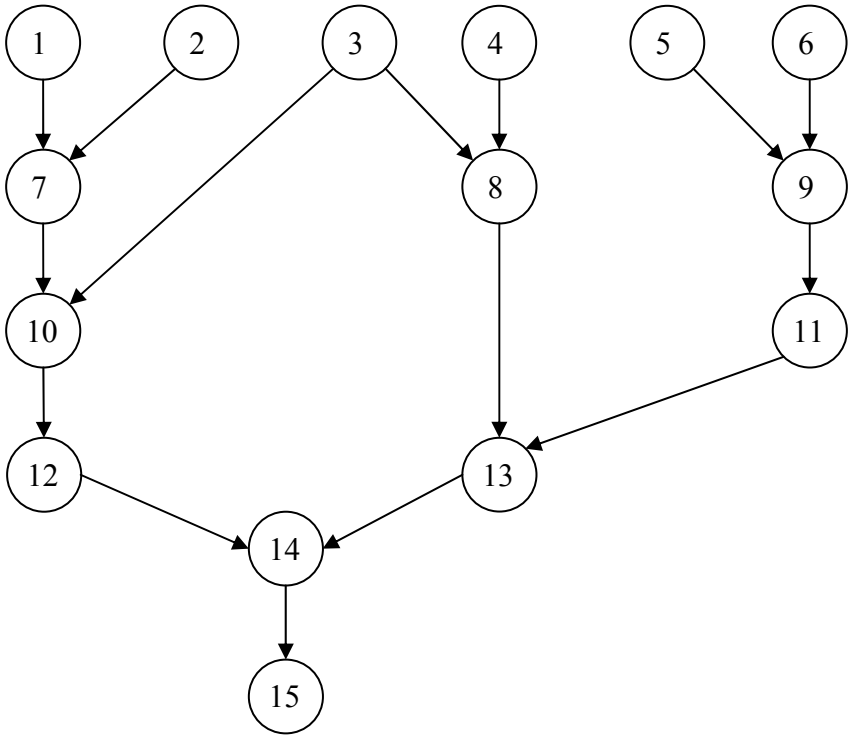

Рис. 1.1. Граф алгоритма реализации математической модели формирования процесса Y

Определим параметры эффективности данного способа представления графа. Согласно определению матрицы инцидентности, потребность в памяти для ее хранения будет равна $Q_M = I \times J$; каждый столбец матрицы содержит один и только один элемент, равный единице, и, аналогично, один элемент, принимающий значение минус единица, т. е. общее количество информативных ячеек равно $2J$. Тогда коэффициент использования памяти определится так:

$$K_J = 2J / (I \cdot J) = 2 / I.$$

Достоинствами такого представления топологии графа являются наглядность и простота дальнейших преобразований графа. Существенные недостатки – потребность в большом объеме памяти и крайне нерациональное ее использование.

Другим распространенным способом описания топологии графа является *матрица смежности*.

Матрицей смежности называется квадратная матрица $n \times n$ с элементами b_{ij}, для которой

$$b_{ij} = \begin{cases} 1, \text{ если из } i\text{-й вершины в } j\text{-ю вершину идет дуга,} \\ 0 \text{ в противном случае.} \end{cases}$$

Данное определение применимо для ориентированных графов. В общем случае используется следующее определение. Матрица смежности есть квадратная матрица размером $n \times n$, где n – число вершин графа, у которой элементы b_{ij} определяются следующим образом:

$$b_{ij} = \begin{cases} 1, \text{ если вершины } v_i \text{ и } v_j \text{ инцидентны,} \\ 0 \text{ в противном случае.} \end{cases}$$

Объем памяти, требуемый для хранения матрицы смежности, в обоих случаях равен $Q_M = I^2$, коэффициент использования памяти – $K_J = J / I^2$ и $K_J = 2J / I^2$ для случаев ориентированных и неориентированных графов соответственно.

Достоинства и недостатки представления те же, что и у матрицы инцидентностей.

Для неориентированных графов может также использоваться список смежности. В этом представлении каждой вершине графа сопоставляется список смежных вершин вида

$$v_i : v_{i_1}, v_{i_2} \ldots v_{i_k},$$

где v_{i_k}, $k = \overline{1, K_i}$ – вершины, смежные с вершиной v_i.

Наглядным является также способ задания графа списком дуг (инцидентором) $e_k = (V, W)$, $k = \overline{1, K}$, где K – общее число дуг графа.

Параметры оценки эффективности для такого представления:

$$Q_M = 2J,\ K_J = 2J / 2J = 1.$$

Достоинство способа – существенное сокращение по сравнению с матричными представлениями затрат памяти. При этом алгоритмы дальнейшей обработки графа могут оказаться менее удобными.

В наиболее простых случаях может применяться вектор смежности. Вектором смежности называют вектор $\boldsymbol{c}$ размерностью I, i-й элемент которого равен индексу вершины, в которую входит дуга, выходящая из i-й вершины.

Ограничение применения данного способа – возможность его использования только для графов, из каждой вершины которых выходит не более одной дуги.

В соответствии с вышесказанным для графа, приведенного на рисунке со спецификацией в таблице, матрица инцидентности имеет вид:

$$A_{15\times15} = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & -1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -1 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -1 & -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & -1 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & -1 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & -1 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1
\end{bmatrix}$$

Матрица смежности для данного примера:

$$B_{15\times15} = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}$$

Список дуг для данного графа следующий:
$e_1 = (1,\ 7)$; $e_2 = (2,\ 7)$; $e_3 = (3,\ 10)$; $e_4 = (3,\ 8)$; $e_5 = (4,\ 8)$; $e = (5,\ 9)$; $e_7 = (6, 9)$; $e_8 = (7, 10)$; $e_9 = (8, 13)$; $e_{10} = (9, 11)$; $e_{11} = (10, 12)$; $e_{12} = (11, 13)$; $e_{13} = (12, 14)$; $e_{14} = (13, 14)$; $e_{15} = (14, 15)$.

Построить вектор смежности для данного примера нельзя.

Отметим, что с точки зрения автоматизации проектирования наиболее удобным из рассмотренных является третий способ описания графа, так как он требует наименьших затрат машинной памяти. Кроме того, спецификацию вершин графа удобно представлять так называемым *вектором спецификации*, под которым будем понимать вектор $\varphi = (\varphi_1, \varphi_{2,\ \ldots,}\ \varphi_I)$, *i*-й элемент которого представляет собой условное обозначение операции, отождествленной с *i*-й вершиной графа.

В дальнейшем под термином «граф» будем понимать ориентированный граф (орграф), т. е. такой граф, в котором направления передачи данных определены и соответствуют направлениям стрелок дуг.

1.1.3. Функциональные устройства. Для реализации модели (1.1) необходимо наличие совокупности *функциональных уст-*

ройств (ФУ), программа работы которых связана с указанием последовательности операций над входными и промежуточными переменными. Под ФУ будем понимать в зависимости от степени детализации алгоритма элементы вычислительной техники (сумматоры, вычислители, устройства для выполнения специальных функций) либо функционально законченные блоки или устройства (унифицированные или специализированные процессоры, сопроцессоры, матричные процессоры и т. д.). Все ФУ делятся на основные и служебные. Основными являются ФУ, которые непосредственно выполняют операции алгоритма согласно математической модели вычислительного процесса (1.1). К служебным ФУ относятся ФУ, не выполняющие непосредственно операций в соответствии с моделью (1.1), но которые необходимы для нормальной организации вычислительного процесса в системе. Примерами таких устройств являются каналы связи, адаптеры, коммутаторы, мультиплексоры, преобразователи уровней сигналов, формирователи импульсных или других типов сигналов и т. д.

По способу организации вычислительного процесса ФУ разделяются на простые и конвейерные. ФУ называется простым, если время выполнения на нем любой операции определено априори и никакая дальнейшая операция не может начать выполняться раньше момента завершения предыдущей операции. ФУ называется конвейерным, если время выполнения на нем любой операции также определено заранее, но дальнейшая операция может начать выполняться через один такт после начала выполнения предыдущей операции.

Исходный граф реализации модели (1.1) будем называть графом вычислительного алгоритма (ГВА).

Реализация модели (1.1) при наличии требований вида (1.2), (1.3) или (1.4) предусматривает решение одной из следующих задач.

Задача синтеза. Задана математическая модель вычислительного процесса или ориентированный граф алгоритма ее реализации. Заданы требования к параметрам времени. Известны характеристики функциональных устройств из заданного набора, способных в совокупности выполнять все операции алгоритма. Заданы показатели качества проектируемой вычислительной системы. Необходимо в рамках заданных ограничений выбрать состав функциональных устройств, построить оргграф, соответствующий вычислительной структуре для реализации заданного алгоритма или их совокупности, разработать программу или множество про-

грамм взаимодействия функциональных устройств, которые бы обеспечивали реализацию алгоритма в реальном времени, выбрать из множества полученных работоспособных структур наилучшим образом удовлетворяющие критериям качества.

Задача анализа. Дана математическая модель вычислительного процесса или ориентированный граф ее реализации, указана спецификация операций, отождествленных с его вершинами. Дан ориентированный граф, описывающий вычислительную структуру, и указана спецификация устройств, отождествленных с его вершинами. Необходимо ответить на вопрос, можно ли реализовать алгоритм на данной структуре, и при положительном ответе указать программу или множество программ, обеспечивающих реализацию алгоритма с учетом требований реального времени.

1.2. Задание

1. Разработать программу формирования графа вычислительного алгоритма для произвольной математической модели.

Исходные данные для программы: математическая модель вычислительного процесса, заданная аналитическим выражением вида (1.1).

Результат выполнения программы – граф вычислительного алгоритма реализации заданной математической модели, представленный:

а) матрицей инцидентностей;

б) матрицей смежности;

в) инцидентором (списком дуг).

2. Продемонстрировать работу программы на тестовом примере, заданном преподавателем.

Лабораторная работа № 2
СИНТЕЗ МАТЕМАТИЧЕСКИХ МОДЕЛЕЙ С ЗАДАННЫМИ СВОЙСТВАМИ

Целью лабораторной работы является изучение метода формирования множества векторов параметров математических моделей с требуемыми свойствами, разработка алгоритма и программы синтеза множества векторов параметров математических моделей с заданными свойствами.

2.1. Задача синтеза математических моделей с требуемыми свойствами

Под задачей синтеза математических моделей вычислительных процессов с требуемыми свойствами будем понимать задачу, заключающуюся в формировании множества усеченных моделей, полученных из базовой модели вычислительного процесса путем выделения ее параметров, которые могут обеспечить требуемые свойства.

Для решения данной задачи предлагается метод, основанный на использовании характеристических логических функций (ХЛФ), связывающих булевы переменные, отождествленные с параметрами модели, и элементов логико-комбинаторного подхода к структурному синтезу сложных систем.

Рассмотрим математическую модель процесса в виде

$$Y = F(\boldsymbol{X}_N), \qquad (2.1)$$

где $\boldsymbol{X}_N$ – вектор параметров модели.

Необходимо найти множество частных (усеченных) моделей

$$Y_y = \{Y_1, Y_2, \ldots, Y_M\},\ Y_{m|m=\overline{1,M}} = F_m(x_\alpha, x_\beta, \ldots, x_\varsigma),\ \alpha, \beta, \ldots, \varsigma \subset [\overline{1, N}],$$

полученных путем выделения тех параметров модели (2.1), которые обеспечивают требуемые свойства.

Свойства модели задаются векторами-строками $\boldsymbol{P}_i$ переменного размера для каждого из элементов вектора-столбца $\boldsymbol{Q}$ классификационных признаков $q_i \leftrightarrow \boldsymbol{P}_i$. Здесь q_i, $i = 1, \ldots, I$ – элементы вектора признаков; p_i^k, $k = 1, \ldots, K_i$ – элементы векторов свойств $\boldsymbol{P}_i$.

Задача синтеза математических моделей с требуемыми свойствами решается в два этапа. На первом этапе формируется множество векторов математических моделей с требуемыми свойствами. В первую очередь из каждого вектора $\boldsymbol{P}_i$ выбирается по одному элементу p_i^k, соответствующему заданному свойству (рис. 2.1).

$\boldsymbol{Q}$

q_1: $\boldsymbol{P}_1 = \left(p_1^1, p_1^2, \ldots, p_1^{K_1}\right)$

q_2: $\boldsymbol{P}_2 = \left(p_2^1, p_2^2, \ldots, p_2^{\mathrm{k}}, \ldots, p_2^{K_2}\right)$

……

q_I: $\boldsymbol{P_I} = \left(p_I^1, p_I^2, \ldots, p_I^{K_I}\right)$

Рис. 2.1. Морфологическая матрица математической модели

В результате расположения выбранных элементов в порядке возрастания индекса i формируется единственный вектор требуемых свойств размерностью I. Например, для морфологической матрицы, изображенной на рис. 2.1, таким вектором будет

$$\boldsymbol{P} = (p_1^1, p_2^k, \ldots, p_I^{K_I}).$$

При этом каждое свойство p_i^k, $k = \overline{1, K}$ обеспечивается включением в выражение модели (2.1) множества параметров

$$X'(i, k)=\left\{x_1'(i, k), x_2'(i, k), \ldots, x_{N_{ik}}'(i, k)\right\},$$

и невключением в это выражение множества параметров

$$X''(i, k) = \left\{x_1''(i, k), x_2''(i, k), \ldots, x_{M_{ik}}''(i, k)\right\}, \left[X'(i, k) \cup X''(i, k)\right] \subset X,$$

где $X = \{x_1, x_2, \ldots, x_N\}$ – множество параметров модели (2.1).

ОПРЕДЕЛЕНИЕ 2.1. Признаки, определяющие отсутствие или наличие ограничений на параметры модели вида

$$(x_\omega)_{\max} = \Omega,\ \omega \neq [1, N],$$

называются *ограничительными*.

Установим взаимообратное соответствие между параметрами модели (2.1) и булевыми переменными $x_n \leftrightarrow y_n = \{0,1\}$.

Введем логические функции вида

$$f_1(i, k) = x_1^B(i, k) \wedge [x_\lambda^B(i, k) \vee x_\eta^B(i, k) \ldots \vee$$
$$\vee (x_\varepsilon^B(i, k) \vee x_\xi^B(i, k)) \vee (x_\chi^B(i, k) \vee x_\varsigma^B(i, k))], \qquad (2.2)$$

если $\left\{x_\iota^B(i,k), x_\lambda^B(i,k), x_\eta^B(i,k), \ldots, x_\varepsilon^B(i,k), x_\chi^B(i,k)\right\} \subset X'(i,k)$,
и

$$f_2 = \overline{x}_\pi^B \wedge \overline{x}_\upsilon^B \wedge \ldots \wedge \overline{x}_\rho^B, \qquad (2.3)$$

если $\{x_\pi, x_\upsilon, \ldots, x_\rho\} \notin X''(i,k)$.

В формулах для $f_1(i, k)$ и $f_2(i, k)$ операция конъюнкции соответствует совместному включению в выражение усеченной математической модели соответствующих параметров, знак дизъюнкции объединяет булевы переменные, отождествленные с параметрами, наличие которых в выражении модели не является необходимым для достижения ее заданных свойств.

ОПРЕДЕЛЕНИЕ 2.2. *Характеристической логической функцией свойства* p_i^k будем называть функцию f_i^k, принимающую значение, равное единице, если модель соответствует требуемым свойствам, и нулю в противном случае.

ЛЕММА 2.1. Характеристическая логическая функция свойства p_i^k определяется следующим образом:

$$f_i^k = f(i,k) \wedge f_2(i,k),$$

где $f(i,k) \wedge f_1(i,k)$, если $X'(i,k) \neq X'(i,l) \cup X'(i,t) \cup \ldots \cup X'(i,q)$,
и

$$f(i,k) = f_1(i,l) \wedge f_1(i,t) \wedge \ldots \wedge f_1(i,q),$$

если $X'(i,k) = X'(i,l) \cup X'(i,t) \cup \ldots \cup X'(i,q) \cup X_p(i,k), X_p'(i,k) \notin$
$\notin X'(i,r), r = \overline{1, K_i}, r \neq k;$

$$\{l, t, \ldots, q\} \subset [1, \ldots, K_i], k \neq l, k \neq t, \ldots, k \neq q,$$

где $f_1(i, k)$ и $f_2(i, k)$ – логические функция вида (2.2) и (2.3); $X'(i, l)$, $X'(i, s)$, …, $X'(i, q)$ – множества параметров, обеспечивающих достижение моделью свойств $p_i^l, p_i^s, \ldots, p_i^q$.

ОПРЕДЕЛЕНИЕ 2.3. *Характеристической логической функцией совокупности свойств* $\left\{p_i^{k_i}\right\}$ называется функция, принимающая значение, равное единице, при наличии у модели всех требуемых свойств, и нулю, если хотя бы одно из свойств отсутствует.

УТВЕРЖДЕНИЕ 2.1. Если $f_1^{k_1}, f_2^{k_2}, \ldots, f_I^{k_I}$ – характеристические логические функции свойств $p_1^{k_1}, p_2^{k_2}, \ldots, p_I^{k_{JI}}$ соответственно, то характеристическая логическая функция набора требуемых свойств имеет вид

$$f\left\{f_i^{k_i}\right\} = f_1^{k_1} \wedge f_2^{k_2} \wedge \ldots \wedge f_I^{k_I}. \qquad (2.4)$$

Действительно, пусть на некотором наборе переменных для всех i и k выполняется $f_i^{k_i} = 1$. Тогда функция f, связывающая аргументы $f_i^{k_i}$ на том же наборе переменных, равняется единице тогда и только тогда, когда все аргументы $f_i^{k_i}$ связаны функцией конъюнкции.

2.2. Задание

1. Разработать программу формирования ХЛФ совокупности требуемых свойств при известных ХЛФ отдельных свойств.

2. Разработать программу определения наборов переменных, при единичном значении которых ХЛФ совокупности свойств принимает единичное значение.

3. Исключить из векторов параметров, полученных при выполнении пункта 2, векторы, для которых ХЛФ отдельных свойств не равны единице.

4. Проверить работу программы на примере, используя табл. 2.1.

Таблица 2.1

Характеристические логические функции заданных свойств

Номер варианта	f_1	f_2	f_3
1	$x_1 \wedge x_2 \wedge (x_4 \vee x_5 \vee \vee x_7) \wedge x_6 \wedge x_8$	$x_4 \vee x_5 \wedge x_8 \wedge (x_1 \vee x_3)$	$x_2 \wedge (x_3 \vee x_5 \vee x_8) \wedge x_7$
2	$x_4 \wedge (x_1 \vee x_6 \vee x_8) \wedge \wedge x_3 \wedge x_5$	$x_2 \wedge x_3 \vee x_7 \wedge (x_1 \vee \vee x_5 \wedge x_7)$	$x_2 \wedge x_4 \wedge (x_4 \vee x_8) \wedge \wedge (x_2 \vee x_5)$
3	$x_1 \wedge x_3 \wedge (x_2 \vee x_5 \vee \vee x_4 \vee x_7)$	$x_3 \wedge x_4 \wedge (x_1 \vee x_6) \wedge x_8$	$x_2 \wedge x_4 \wedge (x_1 \vee x_7 \vee \vee x_3 \vee x_1) \wedge x_6$
4	$x_2 \wedge x_6 \wedge (x_3 \vee x_6) \wedge x_8$	$x_3 \wedge (x_4 \vee x_5) \wedge x_1 \wedge x_7$	$x_1 \wedge x_2 \wedge (x_3 \vee x_6 \vee \vee x_7) \wedge x_8$
5	$x_3 \wedge (x_6 \vee x_7 \vee x_8) \wedge \wedge x_1 \wedge x_4$	$(x_1 \vee x_2 \wedge x_3) \wedge (x_4 \vee \vee x_5 \wedge x_7) \wedge x_8$	$x_2 \wedge x_3 \wedge (x_1 \vee x_7 \wedge \wedge x_6) \wedge x_5$
6	$x_2 \wedge x_3 \wedge x_5 \wedge (x_6 \vee \vee x_1 \vee x_8)$	$x_1 \wedge (x_2 \vee x_6 \wedge x_8) \wedge x_7$	$(x_2 \vee x_1 \wedge x_8) \wedge x_5 \wedge x_4$
7	$x_1 \wedge (x_2 \vee x_4 \vee x_6 \vee \vee x_7) \wedge x_5$	$(x_5 \vee x_3 \wedge x_2) \wedge x_4 \wedge x_6$	$x_2 \wedge x_3 \wedge (x_4 \wedge x_7 \vee x_6)$
8	$(x_1 \vee x_3 \vee x_4) \wedge x_5 \wedge x_7$	$x_4 \wedge (x_6 \wedge x_7 \vee x_3) \wedge x_1$	$x_3 \wedge (x_1 \wedge x_7 \vee x_6) \wedge x_5$

Лабораторная работа № 3

СИНТЕЗ МАТЕМАТИЧЕСКИХ МОДЕЛЕЙ С ТРЕБУЕМЫМИ СВОЙСТВАМИ

Целью лабораторной работы является ознакомление с методами формирования математических моделей с требуемыми свойствами при известных векторах параметров.

3.1. Общие сведения

Непосредственно синтез математических моделей с требуемыми свойствами осуществляется через преобразование графа алгоритма реализации исходной (базовой) математической моделью в соответствии с полученными векторами параметров, обеспечивающих удовлетворение совокупности требуемых свойств. Таким образом, в качестве исходных данных программа синтеза математических моделей вычислительных процессов (моделирующих физические процессы, технические системы) с требуемыми свойствами использует:

а) аналитическое выражение исходного процесса

$$\boldsymbol{Y} = \boldsymbol{F}(\boldsymbol{X}), \tag{3.1}$$

где $\boldsymbol{X}$ – вектор параметров базовой математической модели; $\boldsymbol{Y}$ – вектор выходных данных;

б) векторы параметров математических моделей с требуемыми свойствами

$$\{\boldsymbol{X}_r(\boldsymbol{P}_r)\},$$

где $\boldsymbol{P}_r$ – r-й вектор совокупности требуемых свойств.

УТВЕРЖДЕНИЕ 3.1. Для формирования графа алгоритма реализации математической модели $F(\boldsymbol{x})$ с требуемыми свойствами необходимо над парой вершин u и v графа алгоритма реализации базовой математической модели $G = (V, E)$: $e_0 = (u, v)$, $u \leftrightarrow$ «Ввод x», $x_u \notin \boldsymbol{X}_r(\boldsymbol{P}_r)$, где $\boldsymbol{X}_r(\boldsymbol{P}_r)$ – множество элементов вектора параметров требуемых свойств $\boldsymbol{X}_r(\boldsymbol{P}_r)$, выполнить операцию простого элементарного гомоморфизма (см. л. р. № 10).

Для доказательства рассмотрим подграф $G_k = (V_k, E_k)$ графа G, содержащий вершины u и v и все дуги, инцидентные вершине v – множества входных $E_v^{(\text{вх})} = \{e_1^{(\text{вх})}, e_2^{(\text{вх})}, \ldots, e_I^{(\text{вх})}\} \subset E$ и выходных $E_v^{(\text{вых})} = \{e_1^{(\text{вых})}, e_2^{(\text{вых})}, \ldots, e_J^{(\text{вых})}\} \subset E$ дуг для данной вершины, при этом $e_i^{(\text{вх})} = (u, v)$ (рис. 3.1, *а*). Дуга (u, v) во множестве E соответствует передаче параметра x_u операции, отождествленной с вершиной v. При построении графа $G = (V, E)$ реализации модели $\boldsymbol{F}'(\boldsymbol{X}')$, полученной из модели $\boldsymbol{F}(\boldsymbol{X})$ исключением из нее параметра x_u, соответствующий подграф $G'_k = (V'_k, E'_k)$ содержит вершину v и множество дуг, инцидентных данной вершине:

$$E'_v = E'^{(\text{вх})}_v \cup E'^{(\text{вых})}_v, \; E'_v \leftrightarrow e_v = E_v^{(\text{вх})} \cup E_v^{(\text{вых})} \setminus e_i^{(\text{вх})},$$

где $e_v \in E$, E'_v – множество инцидентных вершине v дуг подграфа G'_k, соответствующих подмножеству u_v вершин подграфа G_k, связанных с вершиной v.

Тогда вершину v во множестве вершин подграфа G'_k можно считать образом пары вершин u и v подграфа G_k, а дуги из подмножества E'_v – образами дуг подмножества e_v подграфа G_k, причем $e_v = E_v \setminus e_i^{(\text{вх})}$. Это значит, что подграф G'_k может быть получен из подграфа G_k преобразованием, которое соответствует операции простого элементарного гомоморфизма (соответствующее преобразование представлено на рис. 3.1, причем $I' = I - 1$, $J' = J - 1$).

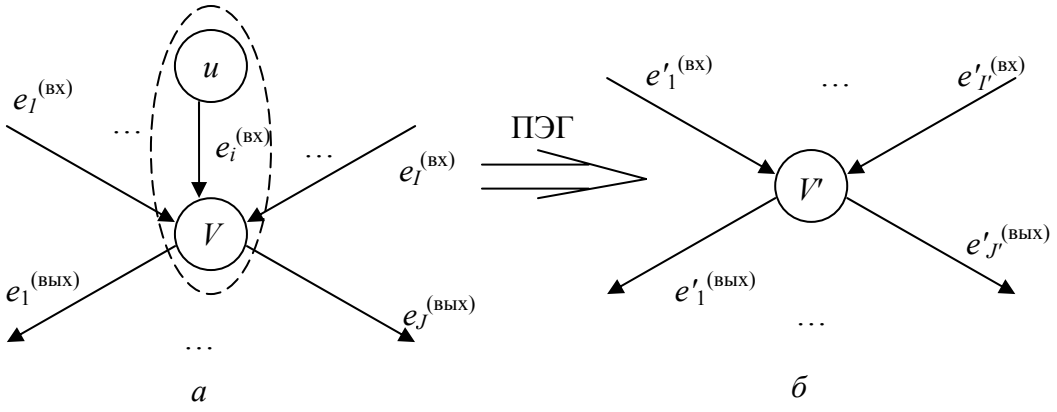

Рис. 3.1. Преобразование подграфа G_k:
а – подграф графа G_k до преобразования; *б* – преобразованный подграф G'_k

УТВЕРЖДЕНИЕ 3.2. Если во множестве вершин V' графа $G' = (V', E')$, полученного путем выполнения простого элементарного гомоморфизма над вершинами графа $G = (V, E)$ алгоритма реализации базовой модели, вершина v' является образом вершин v и u,

и в графе $G = (V, E)$ (u, v) – единственная дуга, входящая в вершину v, то для построения графа алгоритма реализации математической модели с требуемыми свойствами над каждой из пар вершин $v_i', w_{i|i=\overline{1,I}}'$ выполняется операция простого элементарного гомоморфизма, где w_i – i-я вершина, которой поставляет данные вершина v'; I – количество дуг, выходящих из вершины v'.

Действительно, если во множестве вершин графа $G = (V, E)$ вершина v соответствует одноместной операции обработки параметра, поставленного ей вершиной u, то во множестве вершин преобразованного графа $G' = (V', E')$, который не содержит вершины $u' \leftrightarrow u$, образ v' вершины v выполняет «пустую» операцию над отсутствующими данными, поэтому любая из пар вершин $v_i', w_{i|i=\overline{1,I}}'$ свертывается в одну вершину w''_i, которая становится их образом.

Соответствующее преобразование представлено на рис. 3.2.

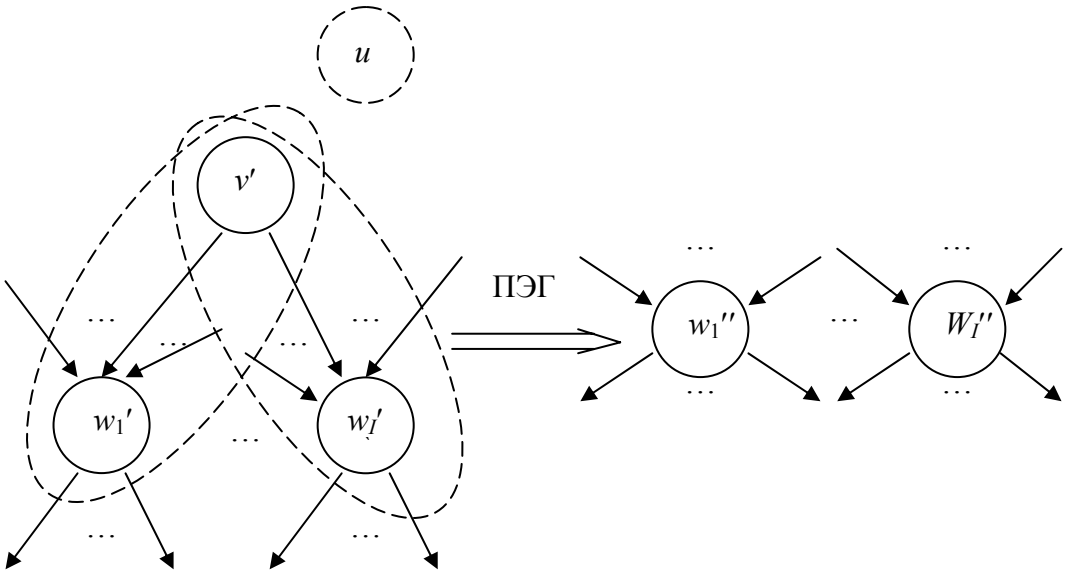

Рис. 3.2. Преобразование подграфа G_k при одноместной операции $\varphi \leftrightarrow u$:
а – подграф после первого преобразования;
б – вторично преобразованный подграф

УТВЕРЖДЕНИЕ 3.3. Если в результате преобразования графа $G = (V, E)$ алгоритма реализации базовой математической модели количество входных дуг вершины v' меньше, чем требуемая вместимость операции $\varphi \leftrightarrow v'$, то для получения графа алгоритма реализации математической модели с требуемыми свойствами необ-

ходимо выполнить операцию простого элементарного гомоморфизма над каждой из пар вершин v', w'_i, $i = \overline{1, I}$, где w'_I и I имеют тот же смысл, что и в предыдущем случае.

Действительно, если в результате преобразования графа G количество дуг, входящих в вершину v', меньше, чем требуемое количество аргументов операции, отождествленной с вершиной $v \in V$, вершина v' выполняет функцию транзита данных с выхода вершины u' каждой из вершин w'_i, $i = \overline{1, I}$ (рис. 3.3).

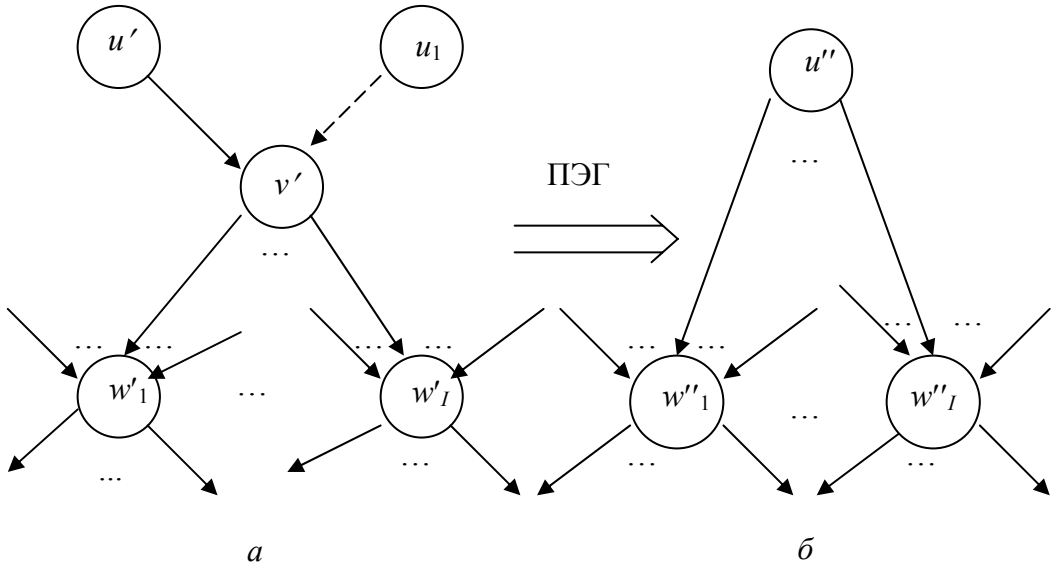

Рис. 3.3. Преобразование подграфа G_k
при использовании операцией $\varphi \leftrightarrow v$ более одного аргумента:
а – подграф G_k после первого преобразования;
б – результат вторичного преобразования подграфа G_k

Тогда каждая из пар дуг (u', v'), (v', w'_i) может быть прообразом одной дуги (u'', w''_i) графа $G'' = (V'', E'')$, полученного на очередном этапе преобразования графа G, что и требовалось показать.

С учетом приведенных положений реализация предложенного метода синтеза математических моделей с требуемыми свойствами предусматриваем выполнение следующей последовательности действий:

1. Формирование множества N-мерных векторов параметров модели с требуемыми свойствами.

В качестве исходных данных процедуры используются: аналитическое выражение исходной математической модели $Y = F(\boldsymbol{X}_n)$, набор характеристических логических функций $\{f_i^k\}$, массив гра-

ничных значений количества свойств для классификационных признаков $\{K_i\}$.

Для реализации процедуры используются определения 2.2, 2.3, лемма 2.1, утверждение 2.3. При этом получаем совокупность $\{\boldsymbol{X}_i(\boldsymbol{P}_r)\}$.

2. Построение графа математической модели вычислительного процесса с требуемыми свойствами.

Исходными данными являются: аналитическое выражение базовой математической модели $Y = F(\boldsymbol{X}_N)$, вектор параметров этой модели $\boldsymbol{X}_N$, векторы параметров с требуемыми свойствами $\{\boldsymbol{X}_r(\boldsymbol{P}_r)\}$.

Процедура выполняется следующими этапами:

а) построение графа G вычислительного алгоритма реализации базовой модели $Y = F(\boldsymbol{X}_N)$;

б) для каждого из векторов $\boldsymbol{X}_r(\boldsymbol{P}_r)$ преобразование графа G в соответствии с утверждениями 3.2–3.3 с целью получения множества графов $\{G_r = (V_r, E_r)\}$ алгоритмов реализации усеченных математических моделей со свойствами $\{P_r\}$; при этом для каждого полученного графа формируется спецификация путем удаления элементов идентификации операций, отождествленных с вершинами, поставляющими данные в каждой паре свертываемых дуг графа G другому прообразу того же образа во множестве вершин V_r;

в) для каждого из графов $\{G_r\}$ обратный переход от топологии и его спецификации к аналитическому выражению математической модели с требуемыми свойствами.

Рассмотрим пример. Математическая модель процесса имеет вид

$$y = E_0 \cdot Z_\text{ф} \cdot F_\text{з}(U(t), (\tau_\text{з} + \tau'_\text{з})) \cdot F_\text{з}(\chi((t_0), \tau_u)\, \tau_\text{з} + \tau'_\text{з})), \qquad (3.2)$$

где E_0, $U(t)$, $\tau_\text{з}$, $\tau'_\text{з}$, τ_u – амплитудные и временные параметры процесса; t_0 – момент начала формирования процесса; $F_\text{з}(\cdot)$ – специальная функция.

Данная модель характеризуется двумя классификационными признаками, каждый их которых принимает два значения (свойства):

$$\boldsymbol{P}_1 = \left(p_1^1, p_1^2\right),$$
$$\boldsymbol{P}_2 = \left(p_2^1, p_2^2\right).$$

Предположим, что характеристические логические функции данных свойств имеют вид:

$$\boldsymbol{P}_1\colon f_1^1 = \left(E_0^B \vee K_E^B \vee Z_\text{ф}^B \vee U^B\right) \wedge \tau_u^B \wedge \left(\tau_\text{з}^B \vee \tau_\text{з}^{'B}\right);$$
$$f_1^2 = \tau_u^B \wedge \left(\tau_\text{з}^B \vee \tau_\text{з}^{'B}\right) \wedge E_0^B \wedge \boldsymbol{K}_E^B \wedge \boldsymbol{Z}_\text{ф}^B \wedge \boldsymbol{U}^B;$$

$$\boldsymbol{P}_2\colon f_2^1 = \tau_u^B \wedge \left(E_0^B \vee Z_\text{ф}^B \vee U^B \vee \tau_\text{з}^B\right) \wedge \boldsymbol{K}_E^B \wedge \boldsymbol{\tau}_\text{з}'^B;$$

$$f_2^2 = \tau_\text{з}'^B \vee Z_\text{ф}^B.$$

Найдем математические модели для процесса, характеризующегося вектором свойств

$$\boldsymbol{P} = \left(p_1^2, p_2^1\right). \tag{3.3}$$

Граф алгоритма реализации модели (3.2) и спецификация его вершин представлены на рис. 3.4 и в табл. 3.1 соответственно.

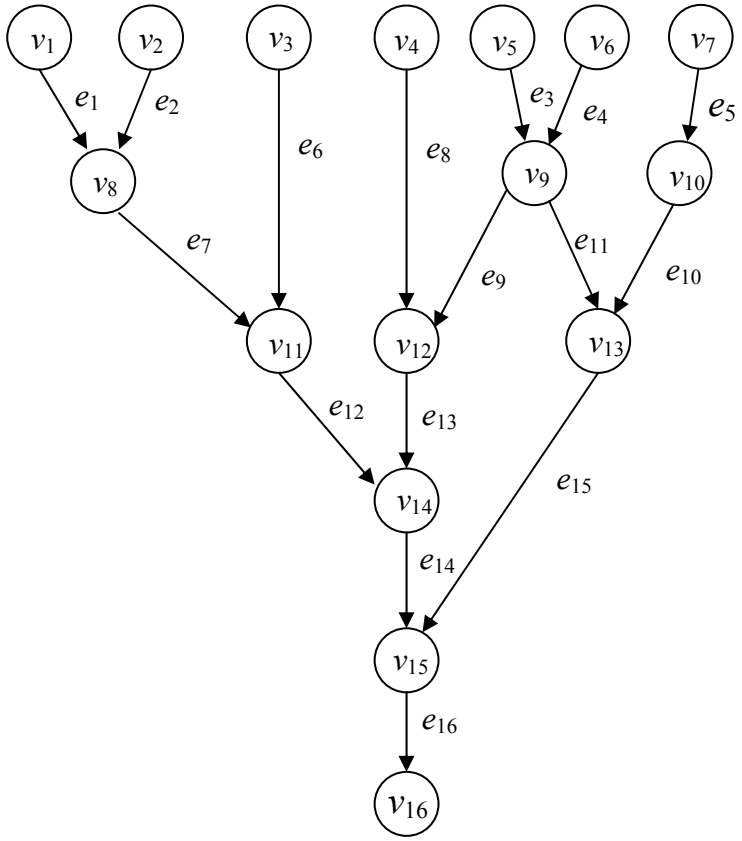

Рис. 3.4. Граф алгоритма реализации базовой модели:
v_1–v_{16} – вершины; e_1–e_{16} – дуги графа

Таблица 3.1

Спецификация графа вершин ВГА реализации базовой модели

v_i	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8
f_j	Ввод E_0	Ввод $Z_\text{ф}$	Ввод K_E	Ввод U	Ввод $\tau_\text{з}$	Ввод $\tau_\text{з}'$	Ввод τ_u	*

Окончание табл. 3.1

V_i	v_9	v_{10}	v_{11}	v_{12}	v_{13}	v_{14}	v_{15}	v_{16}
f_j	+	$\chi_\text{р}$	*	F_3	F_3	*	*	Вывод Y

Используя утверждение 1.3, следствие 1.2, получим вектор параметров модели (3.2) со свойствами (3.3):

$$\boldsymbol{X}_1(N_1) = (\tau_з, \tau_u). \qquad (3.4)$$

Выполнив над графом для рассмотренной модели, вектора (3.4) и вектора параметров базовой модели $\boldsymbol{X}_N = (E_0, Z_ф, K_E, U, \tau_u, \tau_з, \tau'_з)$ действия в соответствии с утверждениями 3.1–3.3, получим граф, изображенный на рис. 3.5 со спецификацией, представленной в табл. 3.2.

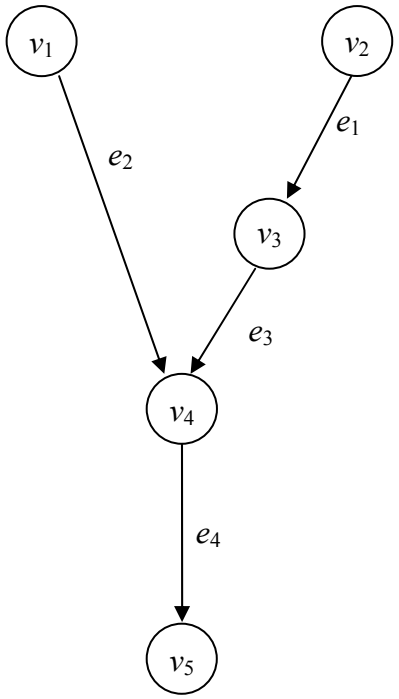

Рис 3.5. Граф алгоритма реализации математической модели с требуемыми свойствами

Таблица 3.2

Спецификация вершин графа ВА реализованной модели с требуемыми характеристиками

φ_i	v_1	v_2	v_3	v_4	v_5
f_j	Ввод $\tau_з$	Ввод τ_u	$\chi_р$	F_3	Вывод Y

На последнем этапе осуществляется переход от графа G'' к аналитическому выражению математической модели, которое принимает вид:

$$y_y^{(1)} = F_з\left(\chi_з\left((t_0), \tau_u\right), \tau_з\right).$$

3.2. Задание

1. Изучить вопросы, связанные с формированием математических моделей с требуемыми свойствами.

2. Разработать алгоритм и программу формирования графа вычислительного алгоритма реализации базовой математической модели.

Исходные данные:

– число параметров модели – до 10;

– операции – произвольные, в том числе и специальные функции;

– способы представления ГВА – список дуг, матрица инцидентности, матрица смежности.

3. Разработать алгоритм и программу преобразования исходного ГВА при заданном векторе параметров математической модели с требуемыми свойствами.

4. Протестировать разработанный программный продукт на примере, заданном преподавателем.

Лабораторная работа № 4

ОПРЕДЕЛЕНИЕ ВСЕХ ПОЛНЫХ ПУТЕЙ ГРАФА ВЫЧИСЛИТЕЛЬНОГО АГОРИТМА

Целью лабораторной работы является изучение методов поиска полных путей графа вычислительного алгоритма реализации математической модели проектируемой системы.

4.1. Понятие полного пути

Понятие *полный путь* не является характерным для терминологии теории графов, однако в рамках теории синтеза вычислительных систем реального времени данное словосочетание является устоявшимся термином.

ОПРЕДЕЛЕНИЕ 4.1. Полным путем L графа алгоритма называется путь, связывающий одну из начальных вершин графа с одной из его конечных (концевых, висячих) вершин.

В соответствии с данным определением множество полных путей представляет собой набор векторов, элементами которых являются индексы последовательно соединенных дугами вершин графа.

Очевидно, что набор всех полных путей графа наряду с методами, описанными в теоретическом материале к первой лабораторной работе, также является одним из способов описания топологии графа. По сравнению с рассмотренными ранее методами данный способ является менее наглядным и удобным для дальнейших преобразований графа.

4.2. Пример

Набор всех полных путей графа, представленного на рис. 1.1, будет следующим: $\boldsymbol{L}_1$ = (1, 7, 10, 12, 14, 15); $\boldsymbol{L}_2$ = (2, 7, 10, 12, 14, 15); $\boldsymbol{L}_3$ = (3, 10, 12, 14, 15); $\boldsymbol{L}_4$ = (3, 8, 13, 14, 15); $\boldsymbol{L}_5$ = (4, 8, 13, 12, 14, 15); $\boldsymbol{L}_6$ = (5, 9,11, 13, 14, 15); $\boldsymbol{L}_7$ = (6, 9, 11, 13, 14, 15).

4.3. Задание

1. Разработать программу формирования всех полных путей графа вычислительного алгоритма для произвольной математической модели.

Исходные данные для программы: граф вычислительного алгоритма.

Результат выполнения программы – множество векторов $\{\boldsymbol{L}\}$ всех полных путей ГВА.

2. Продемонстрировать работу программы на тестовом примере, заданном преподавателем.

Лабораторная работа № 5

НАЗНАЧЕНИЕ УРОВНЕЙ ВРЕМЕННОЙ ИЕРАРХИИ ВЕРШИНАМ ГРАФА ВЫЧИСЛИТЕЛЬНОГО АЛГОРИТМА

Целью лабораторной работы является ознакомление с понятием уровня временной иерархии (УВИ), изучение принципов назначения уровней временной иерархии вершинам ГВА.

5.1. Определение уровня временной иерархии

Для отражения различных требований по скорости обработки данных по различным путям обработки вводится понятие *уровня временной иерархии* (УВИ).

ОПРЕДЕЛЕНИЕ 5.1. Уровнем γ временной иерархии подмножества вершин графа ($G = V, E$) называется их приоритет, соответствующий требуемой скорости обработки данных каждой из вершин данного подмножества и определяемый следующим образом:

$$\forall v_i \in V: v_i \leftrightarrow \Delta t\left(\varphi_i\right) = \Delta t(\gamma) | \gamma = \overline{1, \Gamma};$$

$$\Delta t(1) = \min\left(\Delta t(\varphi \subset \Phi)\right);$$

$$\Delta t(\gamma) = \min\left(\Delta t\left(\varphi \subset \Phi \setminus \bigcup_{j=1}^{\gamma-1} \Phi(j)\right)\right);$$

$$\Gamma \leftrightarrow \max\left(\Delta t(\varphi)\right) \vee C,$$

где $\Phi = X \cup V \cup Z \leftrightarrow V$; $\Phi(\gamma) = \bigcup_{k=1}^{K_\gamma} \varphi_k \leftrightarrow \Delta t(\gamma)$ – подмножества переменных модели (1.1) мощностью K_γ.

ОПРЕДЕЛЕНИЕ 5.2. Подмножество вершин $V(\gamma)$ графа алгоритма $G = (V, E)$, для которого установлено соответствие $V(\gamma) \leftrightarrow \Delta t(\gamma)$, называется множеством вершин γ-го уровня временной иерархии.

ОПРЕДЕЛЕНИЕ 5.3. Интервал времени $\Delta t(\gamma_i)$, для которого установлено соответствие $\Delta t(\gamma) \leftrightarrow V(\gamma)$: $G = G(V, E)$, $V(\gamma) \subset V$, называется шагом дискретизации γ-го уровня временной иерархии.

В соответствии с определениями 5.1–5.3 каждой i-й вершине ГВА ставится в соответствие некоторый интервал времени $\Delta t(\gamma_i)$, задающий периодичность формирования переменной γ_i на выходе i-й вершины (для вершин, отождествленных с операцией ввода данных – периодичность поступления соответствующей переменной на вход системы). Подмножеству вершин ВГА, которым предъявлены одинаковые требования к скорости данных, назначается уровень временной иерархии γ, характеризующийся шагом дискретизации $\Delta t(\gamma)$. Наивысший (первый) УВИ назначается вершинам, требующим максимальной скорости обработки данных. Другие, более низкие, УВИ назначаются вершинам по мере снижения требований к скорости обработки данных (увеличения значения шага дискретизации).

Вершинам, для которых не определены требования по скорости обработки данных, УВИ назначаются исходя из следующего положения.

УТВЕРЖДЕНИЕ 5.1. Вершина v_κ графа алгоритма $G = (V, E)$, для которой $\exists v_k$: v_k Δt_k принадлежит множеству $V(\gamma)$ вершин γ-го уровня временной иерархии, где $v_\kappa \leftrightarrow \min\{\Delta t_i\}$, $\min\{\Delta t_i\} \leftrightarrow \Delta t(\gamma) \leftrightarrow V(\gamma)$; для всех i: $\{a_{ij} = 1\} \rightarrow \{a_{\kappa j} = -1\}$, $i, \kappa = 1, \ldots, I$, $j = 1, \ldots, J$.

Из последнего утверждения, вытекают следующие правила назначения уровней временной иерархии вершинам, для которых не заданы требования по скорости обработки данных:

– если в некоторую вершину ГВА v_j входят дуги, выходящие из вершин v_i и v_k, принадлежащих множествам $V(\alpha)$ и $V(\beta)$ соответственно, и $\alpha < \beta$, т. е. УВИ вершины v_i выше УВИ, назначенного вершине v_k, то вершине v_j назначается уровень вершины v_i, т. е. v_l принадлежит множеству $V(\alpha)$, где $V(\alpha)$, $V(\beta)$ – множества вершин уровня α и β соответственно;

– если в некоторую вершину входит несколько дуг, выходящих из вершин, которым назначены различные УВИ, то этой вершине назначается УВИ, наивысший из УВИ вершин, поставляющих данные этой вершине.

Результаты выполнения данной процедуры наглядно могут быть представлены в виде таблицы, содержащей номера УВИ, индексы вершин, которым назначены данные УВИ, и соответствующие шаги дискретизации. Такой способ использован в примере,

рассмотренном ниже (подраздел 5.2). Альтернативным способом представления назначения УВИ являются множества вершин УВИ вида $\{V(\gamma)\}$. При этом для задания шагов дискретизации соответствующих УВИ используется *вектор шагов дискретизации*.

ОПРЕДЕЛЕНИЕ 5.4. Вектором шагов дискретизации называется вектор $\Delta\boldsymbol{t}$, i-й элемент которого равен шагу дискретизации УВИ $\gamma = i$:

$$\Delta\boldsymbol{t} = (\Delta t(1), \Delta t(2), \ \ldots, \Delta t(\Gamma)). \qquad (5.1)$$

5.2. Пример

Задание. Определить уровни временной иерархии вершин графа алгоритма реализации модели (1.6) при предъявленных требованиях вида:

$$\Delta t(x_1) = 10, \Delta t(x_2) = \Delta t(x_4) = \Delta t(x_5) = 20, \Delta t(x_3) = \Delta t(x_6) = 40. \quad (5.2)$$

Решение. В первую очередь, в соответствии с определением 5.1, назначаем УВИ вершинам графа реализации модели (1.6), представленного на рис. 1.1, для которых шаги дискретизации заданы в явном виде (5.1), т. е. вершинам первого яруса:

$\gamma = 1$: v_1;

$\gamma = 2$: v_2, v_4, v_5;

$\gamma = 3$: v_3, v_6.

Вершинам нижних ярусов УВИ назначается исходя из утверждения 5.1.

Окончательно назначение УВИ вершинам графа алгоритма реализации модели (1.5) при заданных условиях (5.1) представлено в табл. 5.1.

Таблица 5.1

Назначение уровней временной иерархии вершинам ГВА

Номер УВИ (γ)	Номера вершин (i)	Шаг дискретизации ($\Delta t(\gamma)$)
I	1, 7, 10, 12, 14, 15	10
II	2, 4, 5, 8, 9, 11, 13	20
III	3, 6	40

Множества вершин УВИ для данных из табл. 5.1 имеют вид: $V(1) = (1, 7, 10, 12, 14, 15)$; $V(2) = (2, 4, 5, 8, 9, 11, 13)$; $V(3) = (5, 3)$.

Вектор шагов дискретизации:

$$\Delta \boldsymbol{t} = (10, 20, 40). \qquad (5.3)$$

5.3. Задание

1. Разработать программу назначения уровней временной иерархии вершинам ГВА. Исходные данные:

– граф вычислительного алгоритма $G = (V, E)$;

– интервалы, задающие периодичность поступления очередности отсчетов входных переменных: $\{\Delta t\,(x_i)\}$, где $i = 1, \ldots, I$ – индекс входных переменных.

2. Протестировать разработанную программу на примере, заданном преподавателем.

Лабораторная работа № 6

НАЗНАЧЕНИЕ ФУНКЦИОНАЛЬНЫХ УСТРОЙСТВ ВЕРШИНАМ ГРАФА БАЗОВОЙ СТРУКТУРЫ

Целью лабораторной работы является изучение принципов назначения функциональных устройств вершинам исходного графа реализации вычислительного процесса ВС.

6.1. Формирование графа базовой структуры

6.1.1. Определение графа базовой структуры. В первом приближении выберем в качестве графа вычислительной структуры тот же граф $G = (V, E)$, посчитав, что ФУ, помещенное в какую-то вершину графа, может выполнять соответствующую операцию. Будем также считать, что каждое ФУ получает аргументы согласно графу G. При отсутствии какой-нибудь дуги или ее начальной вершины либо отождествлении начальной вершины с операцией ввода соответствующий аргумент берется извне. Такая структура называется базовой и описывается графом базовой структуры.

ОПРЕДЕЛЕНИЕ 6.1. *Графом базовой структуры* (ГБС) называется граф, описывающий процесс обработки данных вычислительной системой в соответствии с графом вычислительного алгоритма и изоморфный этому графу.

Для формирования ГБС необходимо вершинам ГВА поставить в соответствие (назначить) ФУ, реализующие функции (операции), отождествленные с данными вершинами. Назначение ФУ вершинам ГВА производится в два этапа:

– построение матрицы соответствия;

– формирование множества векторов назначения.

6.2.2. Формирование множества векторов назначения. Множество альтернативных вариантов структуры проектируемого объекта может быть неявно задано в виде множества векторов назначения $\{\boldsymbol{R}\}$.

ОПРЕДЕЛЕНИЕ 6.2. *Вектором назначения* называется вектор $\boldsymbol{R} = (r_1, r_2, ..., r_I)$, i-й элемент которого равен номеру функционального устройства (ФУ), назначенного i-й вершине графа.

Здесь $i = 1, \ldots, I$, где I – количество вершин графа вычислительного алгоритма (ГВА) реализации математической модели проектируемого объекта; $r_i = j$ – номер функционального устройства из множества $Z = \{z_1, z_2,..., z_i, ..., z_I\}$ экземпляров классов $Z_{n|n=1,..., N}$ классов типовых элементов, реализующих в совокупности все операции некоторого класса задач, т. е. элемента $z_k^{(n)}$ из подкласса $Z_n \leftrightarrow \varphi_n \leftrightarrow v_i$ класса элементов, реализующих функцию φ_n, отождествленную с вершиной v_i ГВА, N – мощность множества Z_n.

При этом в общем случае можем иметь место пересечения подмножеств (рис. 6.1)

$$z_n : Z = \bigcap_{n-1}^{N} z_n \,. \tag{6.1}$$

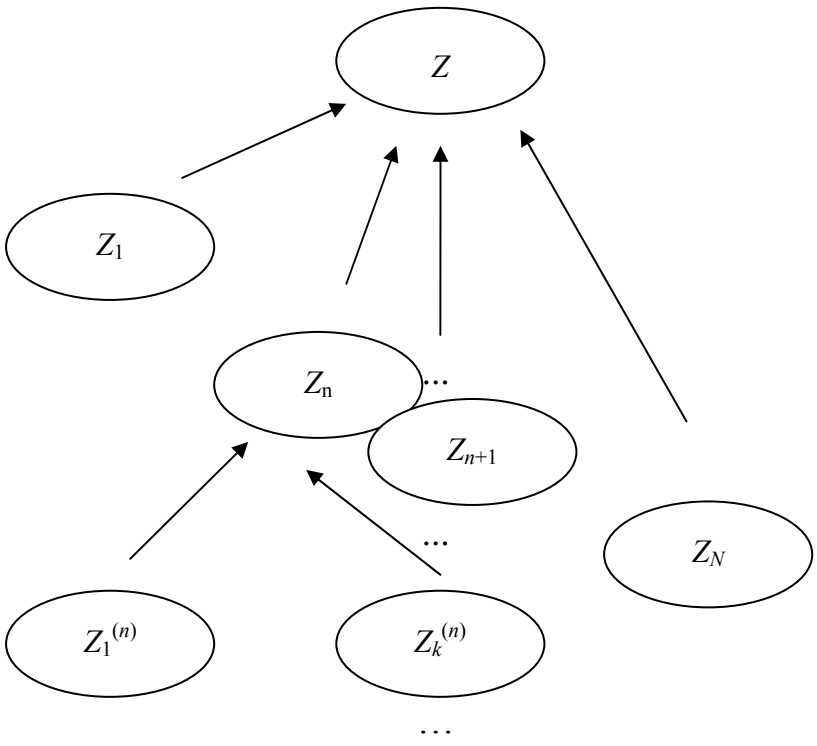

Рис. 6.1. Иерархия технических средств, используемых при синтезе сложного технического оборудования

На данном рисунке $z_k^{(n)}$ – k-й элемент множества Z_n.

Для $z_k^{(n)}$ справедливо

$$Z_n = \bigcup_{k-1}^{K} z_k^{(n)} . \qquad (6.2)$$

Каждый объект $z \in Z$ характеризуется m кортежами вида

$$\langle \tau_n^{(j)}, \alpha_n^{(j)}, \beta_n^{(j)}, ..., \omega_n^{(j)} \rangle, \qquad (6.3)$$

где $\tau_n^{(j)}$ – время реализации j-м ФУ n-й операции, $\alpha_n^{(j)}$, $\beta_n^{(j)}$, $\omega_n^{(j)}$ – атрибуты, соответствующие техническим характеристикам j-го ФУ (потребляемая энергия, стоимость, масса, габариты, надежность и т. п.), учитываемые выборочно в зависимости от постановки задачи проектирования.

Из вышесказанного следует, что назначение ФУ каждой конкретной вершине ГВА определяется неоднозначно из условия удовлетворения определенных требований, среди которых при проектировании ВСРВ первостепенной задачей является удовлетворение требованию реализации вычислительного процесса в реальном времени.

УТВЕРЖДЕНИЕ 6.1. Если вершина v_i принадлежит множеству вершин $V(\gamma) \leftrightarrow \Delta t(\gamma)$, то данной вершине может быть назначено ФУ, для которого выполняется условие

$$\tau_i^j \le \Delta t(\gamma), \qquad (6.4)$$

где τ_i^j – время обработки j-м ФУ операции, отождествленной с вершиной v_i.

Очевидно, что неоднозначность отношения

$$Z_k^{(n)}: \tau_i^{(n)} \le \Delta t_{\text{зад}} \leftrightarrow v_i \qquad (6.5)$$

предполагает множественность вариантов назначения ФУ вершинам ГВА, которая может быть оценена множеством векторов назначения, формируемых из матрицы соответствия.

ОПРЕДЕЛЕНИЕ 6.3. Прямоугольная матрица D размером $I \times J$, где I – множество вершин графа базовой структуры, J – множество ФУ из заданного набора, для которой:

$$d_{ij} = \begin{cases} 1, \text{ если } j\text{-е ФУ реализует операцию,} \\ \text{отождествленную с } i\text{-й вершиной графа;} \\ 0 \text{ во всех остальных случаях,} \end{cases}$$

называется матрицей соответствия.

Реализуемость некоторым ФУ соответствующей операции означает не только возможность выполнения данным ФУ этой операции, но и выполнение условия (6.4).

ОПРЕДЕЛЕНИЕ 6.4. Множество $\boldsymbol{R} = \{\boldsymbol{R}^{(1)}, \boldsymbol{R}^{(2)}, \ldots, \boldsymbol{R}^{(W)}\}$ векторов размером I, элементы которых $r_i^{(w)} = j$: $d_{ij} = 1$, $j = \overline{1, J}$, $i = \overline{1, I}$, $w = \overline{1, W}$; $\boldsymbol{R}^{(n)} \neq \boldsymbol{R}^{(m)}$ $\forall$ $n \neq m$; $n, m = \overline{1, W}$, называется *множеством векторов назначения*.

Мощность этого множества определяется как NP-полный перебор всех комбинаций функциональных устройств, удовлетворяющих требованию реализации процесса в реальном времени:

$$\mathrm{W} = [R] = \prod_{i=1}^{I} \sum_{j=1}^{J} d_{ij}, \qquad (6.6)$$

где d_{ij} – элемент матрицы соответствия; $[R]$ – мощность множества векторов назначения.

Каждому вектору назначения соответствует свой вектор реализации, элементы которого $\{r_k\}$ формируются как параметры τ_i^j соответствующих ФУ.

ОПРЕДЕЛЕНИЕ 6.5. Вектором реализации называется вектор-строка $\boldsymbol{\tau} = (\tau_1, \tau_2, \ldots, \tau_K)$, где τ_k, $k = 1, \ldots, K$ – время выполнения операции, отождествленной с вершиной, находящейся в начале k-й дуги.

Из данного определения следует, что каждому из найденных векторов назначения соответствует свой уникальный вектор реализации. Тогда каждый из априорных альтернативных вариантов может быть описан кортежем вида

$$\mathrm{АР0}_w = \{G^{\mathrm{БС}}(V^{\mathrm{БС}}, E^{\mathrm{БС}}), \{\varphi_i(v_i)\}, \boldsymbol{R}_w, \boldsymbol{\tau}_w\}, \qquad (6.7)$$

где $\mathrm{АР0}_w$ – априорное решение нулевого уровня для w-го вектора назначения; $G^{\mathrm{БС}}$ – граф базовой структуры; $\{\varphi_i(v_i)\}$ – спецификация вершин ГБС; $\boldsymbol{R}_w$ и $\boldsymbol{\tau}_w$ – w-й вектор назначения и вектор реализации соответственно.

6.2. Примеры

Задание 1. Построить множество векторов назначения для графа, представленного на рис. 1.1 со спецификацией вершин, заданной табл. 1.1 при временных ограничениях вида (5.1) и характеристиках ФУ, описанных в табл. 6.1. Данная таблица в столбцах, обозначенных как тип операции, содержит временные показатели соответствующих ФУ – τ_i^j, представленные в некоторых абстрактных единицах времени – наносекунды, микросекунды, и т. д.;

ξ – некоторый технический параметр ФУ, выраженный также в абстрактных единицах измерениях соответствующего параметра.

Таблица 6.1

Характеристики ФУ

Номер ФУ	Тип операции								ξ
	+	*	f_1	f_2	*Xp*	*Mx*	Ввод	Вывод	
1	2								40
2	12								25
3	30								15
4		30							60
5		15							80
6		8							120
7			1						150
8			11						70
9			23						60
10				2					25
11				16					20
12				30					15
13					2				100
14					20				50
15					25				40
16						1			10
17						15			2
18							16		10
19							1		5
20								1	8
21	1								10

Решение. Назначение УВИ вершинам ГВА в соответствии с требованиями реального времени, заданными ограничениями (6.1), отражено в табл. 5.1. Матрица соответствия для этого случая будет иметь вид, представленный в табл. 6.2.Для данной матрицы можно построить $W = 1 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 3 \cdot 1 \cdot 2 \cdot 2 \cdot 1 \cdot 2 \cdot 2 \cdot 1 =$ = 6144 векторов назначения:

$$\boldsymbol{R}_1 = (19, 18, 18, 18, 18, 18, 7, 5, 1, 7, 7, 10, 14, 1, 20);$$

...

$$\boldsymbol{R}_{21} = (19, 19, 18, 18, 18, 18, 7, 5, 1, 7, 7, 10, 14, 1, 20);$$

…

$$\boldsymbol{R}_{6144} = (19, 19, 19, 19, 19, 19, 21, 6, 21, 10, 8, 7, 11, 21, 20).$$

Таблица 6.2

Соответствие ФУ вершинам ГВА

№	Номер ФУ (j)																				
$v(_i)$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
1																			1		
2																		1	1		
3																		1	1		
4																		1	1		
5																		1	1		
6																		1	1		
7	1																				1
8					1	1															
9	1	1																			1
10										1											
11							1	1													
12							1														
13										1	1										
14	1																				1
15																				1	

Задание 2. Для одного из найденных в примере 1 векторов назначения сформировать вектор реализации.

Решение. Выберем из множества векторов $\boldsymbol{R}$ вектор, наилучшим образом удовлетворяющий требованию оптимальности по критерию $\xi \to \min$:

$$\boldsymbol{R} = (19, 19, 19, 19, 19, 19, 21, 5, 2, 10, 8, 7, 11, 21, 20).$$

Вектору $\boldsymbol{R}$ соответствует вектор реализации

$$\boldsymbol{\tau} = (1, 1, 1, 1, 1, 1, 1, 1, 15, 12, 2, 11, 1, 16, 1, 1).$$

6.3. Задание

1. Разработать программу формирования множества векторов назначения при следующих исходных данных:

– топология ГВА $G = (V, E)$;

– характеристики функциональных устройств $\{\boldsymbol{\tau}_i^j\}$;

– вектора множества вершин временной иерархии $\boldsymbol{V}(\gamma) = (v_1(\gamma), \ldots, v_N(\gamma))$;

– шаги дискретизации $\{\Delta t(\gamma)\}$.

2. Разработать программу формирования вектора реализации для выбранного вектора назначения при следующих исходных данных:

– вектор назначения $\boldsymbol{R}$;

– характеристики ФУ $\{\boldsymbol{\tau}_i^j\}$.

3. Протестировать разработанные программы на примерах, заданных преподавателем.

Лабораторная работа № 7

ФОРМИРОВАНИЕ ГРАФА БАЗОВОЙ СТРУКТУРЫ С БУФЕРНОЙ ПАМЯТЬЮ

Целью лабораторной работы является изучение правил добавления в $G^{БС}$ вершин для буферизации.

7.1. Для чего нужна буферная память

Необходимость во введении в систему буферной памяти обусловлена возможностью слияния путей обработки данных с различными скоростями этой обработки. Например, для вершин фрагмента графа, изображенного на рис. 7.1, заданы шаги дискретизации $\Delta t(v_i) = \Delta t(v_k) = 1$ у. е. в., $\Delta t(v_j) = 4$ у. е. в., где у. е. в. – условная единица времени.

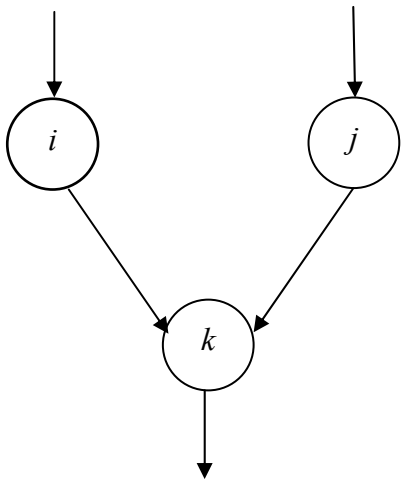

Рис. 7.1. Фрагмент графа

На выходе вершин v_i и v_j формируются данные z_i и z_j соответственно с периодичностью, определенной интервалами $\Delta t_i = \Delta t(v_i) = 1$ и $\Delta t_j = \Delta t(v_j) = 4\Delta t_i$. На выходе вершины v_k формируется результат выполнения операции $z_k = \varphi(z_i, z_j)$ с периодичностью, определяемой интервалом $\Delta t_k = \Delta t_i = 1$ (рис. 7.2), так как при заданных условиях в соответствии с утверждением 5.1 вершине v_k должен быть назначен уровень временной иерархии i-й вершины.

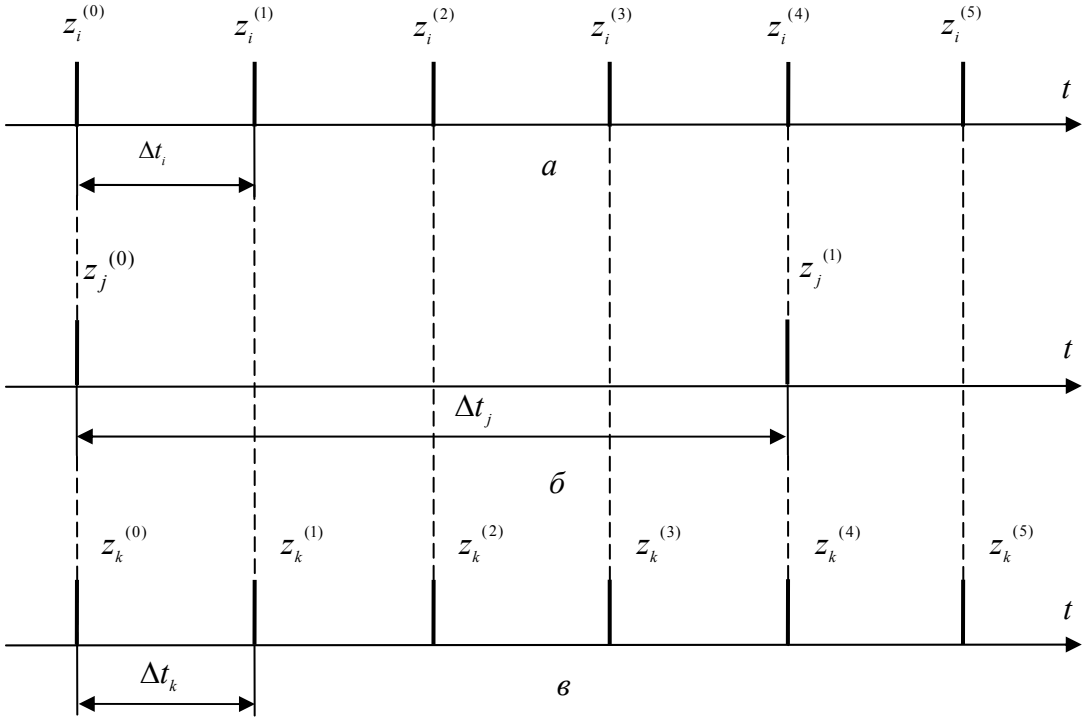

Рис. 7.2. Временная диаграмма формирования результата вершиной v_k:
а – данные на выходе вершины v_i; *б* – данные на выходе вершины v_j;
в – данные на выходе вершины v_k

На рис. 7.2 $z_i^{(0)}, z_i^{(1)}, z_i^{(2)}, \ldots$ − очередные значения промежуточной переменной z_i; $z_j^{(0)}, z_j^{(1)}, z_j^{(2)}, \ldots$ − последовательные значения промежуточной переменной z_j; $z_k^{(0)}, z_k^{(1)}, z_k^{(2)}, \ldots$ − последовательности значений результатов выполнения функции $z_k^{(0)} = \varphi(z_i^{(0)}, z_j^{(0)})$, $z_k^{(1)} = \varphi(z_i^{(1)}, z_j^{(0)})$, $z_k^{(2)} = \varphi(z_i^{(2)}, z_j^{(0)})$, $z_k^{(3)} = \varphi(z_i^{(3)}, z_j^{(0)})$, $z_k^{(4)} = \varphi(z_i^{(4)}, z_j^{(1)})$, $z_k^{(5)} = \varphi(z_i^{(5)}, z_j^{(1)})$ и т. д.

Таким образом, каждое значение данных, полученных в результате выполнения операции, отождествленной с вершиной v_j, по четыре раза используется для формирования очередных значений результатов выполнения операции $\varphi_k(z_i, z_j)$. Это значит, что очередное значение z_j должно быть сохранено на протяжении интервала $\Delta t_j = 4\Delta t_i$. Последнее обстоятельство требует использования памяти, т. е., в общем случае, введения в вычислительную систему блока памяти на выходе ФУ, назначенного j-й вершине, если это ФУ не обладает собственной памятью.

Отметим, что на рис. 7.2, *в* не отображена задержка τ_k, требуемая на выполнение операции φ_k.

7.2. Граф базовой структуры с буферной памятью

Реализация процедуры формирования графа базовой структуры с буферной памятью базируется на следующих положениях.

ОПРЕДЕЛЕНИЕ 7.1. *Графом базовой структуры с буферной памятью* (ГБС БП) называется граф, сформированный из ГБС путем последовательного выполнения операции добавления вершины для дуг, связывающих вершины с разными уровнями временной иерархии.

Каждой из данных вершин назначается блок памяти из условия $\tau_{д}^{j} \le \Delta t(\gamma)$, где $\tau_{д}^{j}$ – время доступа (цикл запись – чтение) j-го ФУ памяти:

$$\Delta t(\gamma) = \min_{i,j} \left\{ \Delta t\left(\gamma_i\right), \Delta t\left(\gamma_j\right) \right\}.$$

Обозначим, в отличие от ГБС, ГБС БП как $G^{БП}$. Для этого графа формируется спецификация путем расширения спецификации ГБС; каждая новая вершина отождествляется с операцией хранения. Строятся также новый вектор назначения $\boldsymbol{R}^{БП}$ и новый вектор реализации $\boldsymbol{\tau}^{БП}$ путем введения соответствующих элементов в векторы $\boldsymbol{R}$ и $\boldsymbol{\tau}$ ГБС.

УТВЕРЖДЕНИЕ 7.1. Если для дуги (v_i, v_j) графа вычислительного алгоритма справедливо $v_i \in V(\gamma_i)$, $v_j \in V(\gamma_j)$, $v_i \upharpoonright v_j$, то для данной дуги должна быть выполнена операция добавления вершины, которая отождествляется с операцией хранения данных.

УТВЕРЖДЕНИЕ 7.2. Вершине, отождествленной с операцией хранения данных и инцидентной двум вершинам v_i и v_j с разным уровнем временной иерархии, должен быть назначен уровень временной иерархии, соответствующий наивысшему из уровней временной иерархии вершин v_i и v_j.

Это требование следует из потребности во входных данных для вершины, в которую входит дуга из данной вершины, с периодичностью, соответствующей шагу дискретизации этой вершины.

Таким образом, в процессе реализации процедуры выполняется преобразование

$$G^{БС} \xrightarrow{ДВ} G^{БП},$$

где ДВ – условное обозначение операции добавления вершины; $G^{БП}$ – граф алгоритма с буферной памятью.

7.3. Пример

Для ГБС, полученного из ГВА, изображенного на рис. 1.1, и УВИ его вершин, соответствующих данным из табл. 5.1, ГБС БП имеет вид, представленный на рис. 7.3, спецификация вершин v_1–v_{15} соответствует табл. 1.1, вершины v_{15}–v_{20} отождествляется с операцией хранения данных.

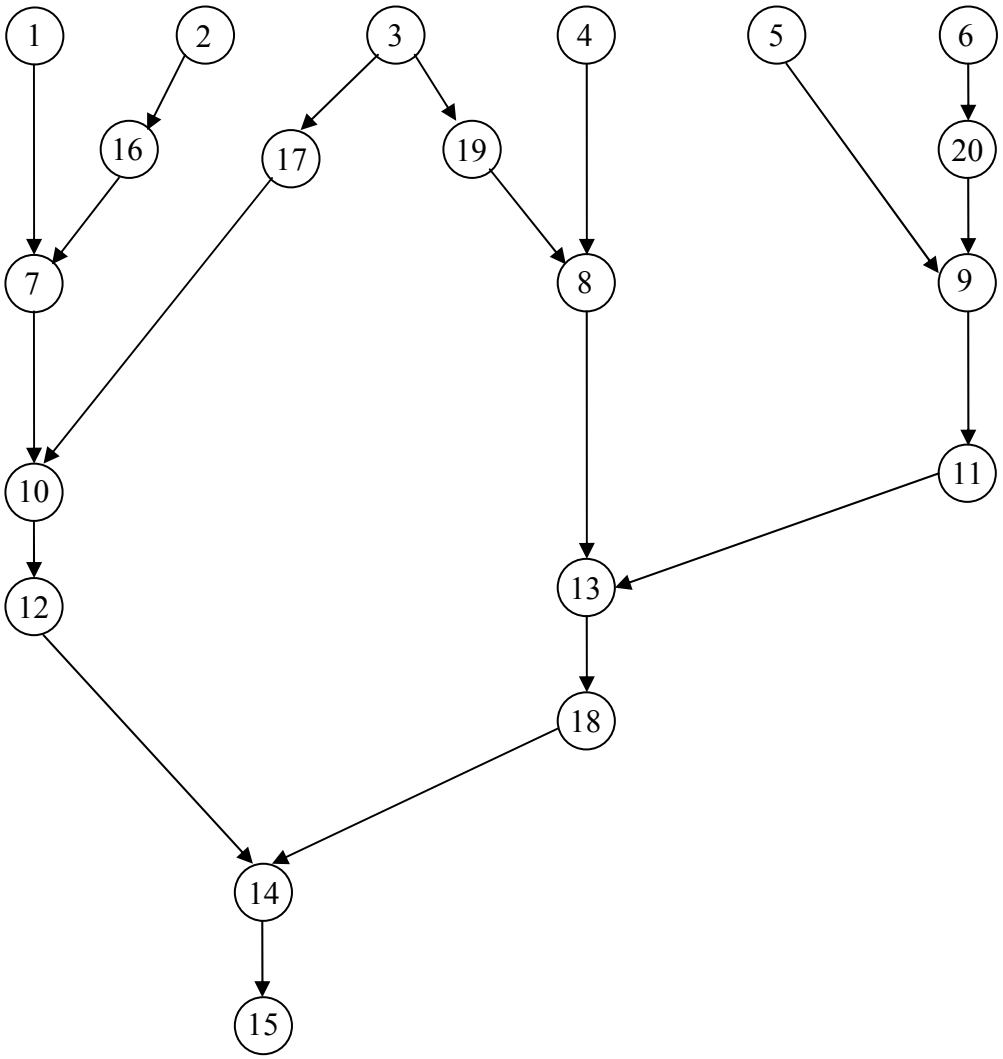

Рис. 7.3. Граф базовой структуры с буферной памятью

Вектор назначения для данного графа получим из вектора назначения (6.8):

$$\boldsymbol{R}^{\text{БП}} = (19, 19, 19, 19, 19, 19, 21, 5, 2, 10, 8,$$
$$7, 11, 21, 20, 13, 13,13, 13, 13). \qquad (7.1)$$

Вектор реализации:

$$\boldsymbol{\tau}^{БП} = (1, 1, 1, 1, 1, 1, 1, 1, 15, 12, 2,$$
$$11, 1, 16, 1, 1, 2, 2, 2, 2). \qquad (7.2)$$

7.4. Задание

2. Разработать программу формирования графа базовой структуры с буферной памятью при следующих исходных данных:

– топология графа базовой структуры $G^{БС}$;

– множества вершин временной $\{V(\gamma)\}$;

– значения шагов дискретизации $\{\Delta t(\gamma)\}$.

3. Протестировать программу на примере, заданном преподавателем.

Лабораторная работа № 8

ФОРМИРОВАНИЕ УСЕЧЕННЫХ ПУТЕЙ

Целью лабораторной работы является изучение принципов формирования усеченных путей γ-х уровней временной иерархии.

8.1. Понятие усеченного пути

8.1.1. Определение усеченного пути. Назначение УВИ вершиной графа предполагает разбиение этого графа, в данном случае ВГАБП, на подграфы в соответствии со следующими положениями.

УТВЕРЖДЕНИЕ 8.1. Для графа алгоритма $G = (V, E)$, вершинам которого назначена временная иерархия, имеет место разбиение

$$G = \bigcup_{q=1}^{Q} G_q,\ G_q = (V_q, E_q);\ \forall v \in V_q \subset V(\gamma): \forall (i,\ j) \subset$$

$$\subset q: \left(\overline{\exists} v: (v \in V_i) \wedge \left(v \in V_j\right)\right)$$

$$\overline{\exists} e: (e \in E_i) \wedge \left(e \in E_j\right);\ \text{при этом } Q \le \Gamma.$$

Справедливость утверждения очевидна, так как если для подмножества вершин

$$v_q \in V(\gamma):\ \forall v \in V(q) \exists \left(e = (v, u) \vee e' = \left(v', u'\right)\right),$$

$$\left(u \vee u'\right) \in V_q,\ \left(e \vee e'\right) \in E_q,$$

то подмножества V_q и E_q образуют подграф $G_q = (V_q, E_q)$, который обрабатывает данные со скоростью, соответствующей $\Delta t(\gamma)$.

Здесь q – номер подмножества вершин соответствующего УВИ ($\gamma = q$).

Для решения проблемы выявления способа отображения ветвей графа на вычислительные архитектуры (конвейерный либо бесконвейерный – см. л. р. № 9) введем понятия усеченного пути (УП) γ-го уровня временной иерархии.

ОПРЕДЕЛЕНИЕ 8.1. *j-м усеченным путем γ-го уровня временной иерархии* $L_j(\gamma)$ называется путь γ-го уровня временной иерархии, для множества вершин V которого справедливо:

$$V(j, \gamma) = V_L(\gamma) \setminus \bigcup_{k=1}^{j-1} V(k, \gamma).$$

В соответствии с данным определением, усеченный путь γ-го УВИ содержит только вершины γ-го УВИ и не должен содержать вершины, включенные в уже сформированные УП, в том числе и пути уровня γ.

По аналогии с полными путями усеченные пути представляются в виде векторов:

$$\boldsymbol{L}_j(\gamma) = (h, \ldots, g),$$

где $h, \ldots, g$ – индексы последовательно соединенных вершин усеченного пути $\boldsymbol{L}_j(\gamma)$.

8.1.2. Формирование усеченных путей. В качестве отправной точки формирования УП будем использовать понятие базисного пути.

ОПРЕДЕЛЕНИЕ 8.2. *Базисным путем* называется путь L_B, максимальный по времени реализации из множества полных путей первого уровня временной иерархии $L^{\text{п}}(1)$ (полного пути, все вершины которого принадлежат первому УВИ) при отсутствии циклов у этих путей или наибольший из циклов первого уровня временной иерархии $L_{\text{ц}}(1)$:

$$L_B = \left\{L^{\text{п}}(1),\ L_{\text{ц}}(1)\right\}: L_B \leftrightarrow \text{Т}(L_B),$$

$$T(L_B) = \begin{cases} \max\limits_{k} \{T_k^{\text{п}}(1)\}, & k = \overline{1, K_1}, \\ \max\limits_{n} \{T_n^{\text{ц}}(1)\}, & n = \overline{1, N_1}, \end{cases}$$

$$V_1(1) \cap V_2(1) \ldots \cap V_k(1) = V(1),$$

где $T_k^{\text{п}}(1)$ – время реализации k-го пути первого уровня временной иерархии; $T_n^{\text{ц}}(1)$ – время реализации наибольшего из циклов для путей первого уровня временной иерархии; $V_k(1) | k = \overline{1, K_1}$ – множество вершин k-го пути первого уровня временной иерархии; K_1 – количество полных путей первого уровня временной иерархии; N_1 – количество циклов для путей первого уровня временной иерархии.

Таким образом, базисный путь – это самый длинный из путей подграфа G_1 (см. утверждение 8.1). Базисный путь переносится во множество усеченных путей γ-х УВИ без преобразований и принимается за первый путь первого УВИ:

$$L_1(1) = L_B.$$

Другие усеченные пути формируются по мере снижения их УВИ (увеличения значения γ) посредством удаления из соответствующих полных путей вершин, включенных в уже сформированные УП, в том числе и путей того же УВИ, что и формируемый, а также вершин более низкого уровня.

8.2. Пример

Задание. Определить усеченные пути для графа, изображенного на рис. 7.3, при извечном векторе назначения (6.8) и уровнях временной иерархии, заданных множеством (5.2).

Решение. Подграф G_1 имеет вид, представленный на рис. 8.1.

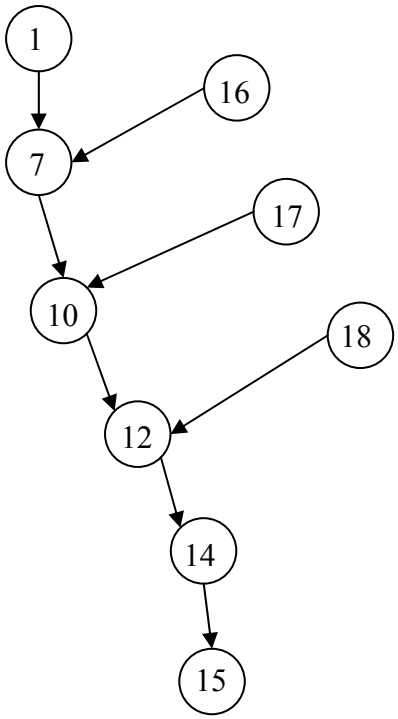

Рис. 8.1. Подграф первого УВИ

В соответствии с определением 8.2 базисный путь будет следующим:

$$\boldsymbol{L}_{B} = (1, 7, 10, 12, 14, 15).$$

Тогда получим:

$$\begin{gathered}\boldsymbol{L}_1(1) = \boldsymbol{L}_B = (1, 7, 10, 12, 14, 15);\ \boldsymbol{L}_2(1) = (16);\\ \boldsymbol{L}_3(1) = (17);\ \boldsymbol{L}_4(1) = (18);\ \boldsymbol{L}_2(1) = (16);\\ \boldsymbol{L}_1(2) = (2);\ \boldsymbol{L}_1(2) = (4, 8, 13);\ \boldsymbol{L}_3(2) = (5, 9, 11);\\ \boldsymbol{L}_4(2) = (19);\ \boldsymbol{L}_5(2) = (20);\ \boldsymbol{L}_1(3) = (3);\\ \boldsymbol{L}_2(3) = (3);\ \boldsymbol{L}_3(3) = (6).\end{gathered} \qquad (8.1)$$

8.3. Задание

1. Разработать программу формирования усеченных путей γ-х уровней временной иерархии для ВГАПБ при известных УВИ его вершин.

2. Проверить корректность программы на примере, заданном преподавателем.

Лабораторная работа № 9
ОПРЕДЕЛЕНИЕ КОНВЕЙЕРИЗИРУЕМЫХ ПУТЕЙ

Целью лабораторной работы является изучение принципов конвейеризации путей графа.

9.1. Принцип конвейеризации в вычислительной технике

9.1.1. Принцип конвейеризации. *Конвейеризация* – это такой метод организации вычислительного процесса, в результате применения которого в вычислительной системе обеспечивается совмещение разных действий по выполнению базовых функций путем их разбиения на подфункции. При этом за основу берутся следующие принципы:

– выполнение функции эквивалентно некоторой последовательности выполнения подфункций;

– данные, являющиеся входными для любой подфункции, являются выходными данными для предыдущей подфункции;

– никаких других взаимосвязей, кроме обмена входными и выходными данными, между подфункциями нет;

– все подфункции могут быть реализованы аппаратными блоками;

– интервалы времени, необходимые для реализации этими аппаратными блоками своих подфункций, примерно равны.

ОПРЕДЕЛЕНИЕ 9.1. Аппаратные средства, необходимые для выполнения любой из этих подфункций, называются *ступенью конвейера*. На вход ступени конвейера данные поступают в дискретные моменты времени.

ОПРЕДЕЛЕНИЕ 9.2. Интервал времени между двумя соседними дискретами загруженного конвейера называется *циклом конвейера*.

9.1.2. Конвейеры операций. Продемонстрируем рассмотренный принцип на примере конвейеризации арифметических операций. Выполнение любой операции складывается из нескольких последовательных этапов, каждый из которых может выполняться своим функциональным узлом. Это легко показать на операциях сложения и умножения. Выполнение деления мантисс

(порядки вычитаются) чаще всего производятся с помощью вычитания из делимого делителя, сдвига влево полученного остатка, нового вычитания делителя из результата сдвига и т. д.

Пусть задана операция, выполнение которой разбито на n последовательных этапов. Пусть t_i – время выполнения i-го этапа. При последовательном их выполнении операция выполняется за время

$$t_{\text{посл}} = \sum_{i=1}^{n} t_i \, [c], \tag{9.1}$$

а быстродействие ЭВМ или одного процессора ВС, выполняющего только эту операцию, составит

$$S_{\text{посл}} = \frac{1}{t_{\text{посл}}} = \frac{1}{\sum_{i=1}^{n} t_i}. \tag{9.2}$$

Выберем время цикла – величину $\tau_k = \{\max t_i\}$, и потребуем при разбиении на этапы, чтобы для любого $i = 1, ..., n$ выполнялось условие $t_i + t_{(i+1) \bmod n} > t_T$. Т. е. чтобы никакие два последовательных этапа (включая конец и новое начало операции) не могли быть выполнены за время одного такта.

Функциональные узлы, выполняющие последовательные этапы одной операции, целесообразно выстроить в единую конвейерную линию, где устройство, выполняющее некоторый этап, закончив его для операции над одним набором данных, переходило бы в следующем такте к выполнению этого же этапа той же операции для другого набора исходных данных.

Например, на рис. 9.1 представлен конвейер выполнения операции сложения.

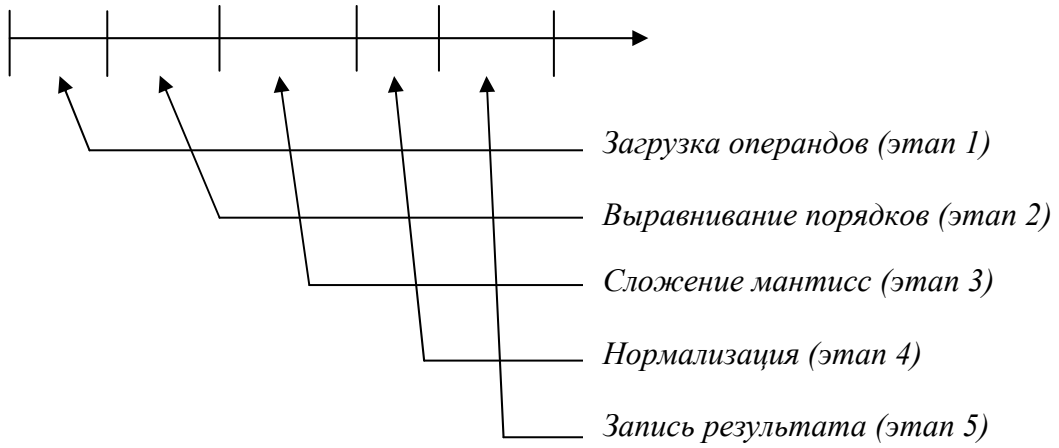

Рис. 9.1. Выполнение операции сложения на конвейере

Пусть реализуется поток команд одного процессора или существует доступ к этому устройству нескольких процессоров так, что в каждом такте возможно задание на выполнение сложения новой пары чисел. Тогда временная диаграмма работы конвейера может иметь вид, представленный на рис. 9.2.

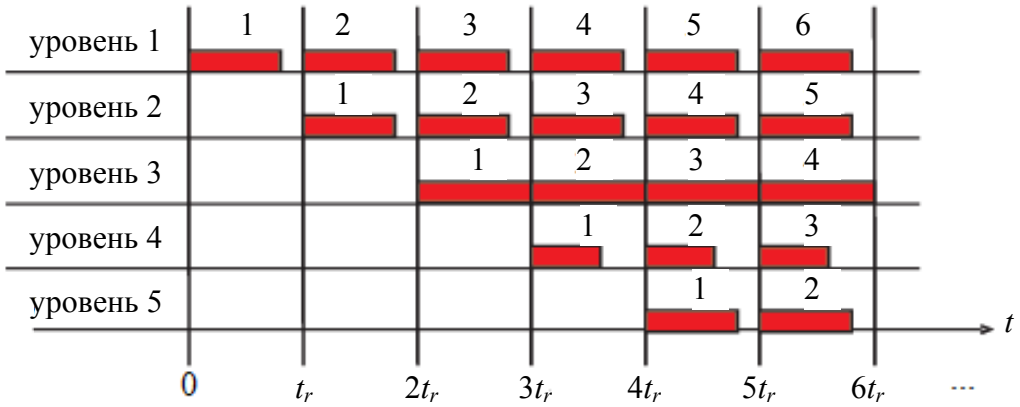

Рис. 9.2. Схема заполнения конвейера

Максимальное быстродействие процессора при полной загрузке конвейера составляет

$$S_{\text{кон}} = \frac{\rho}{t_r} \text{ [операций/с]}. \qquad (9.3)$$

Число ρ – количество уровней (ступеней) конвейера, или глубина перекрытия, Оценка производительности конвейера (9.3) обусловлена тем, что каждый такт на конвейере параллельно выполняются ρ операций. Чем больше число уровней (ступеней), тем больший выигрыш в быстродействии может быть получен.

Известна оценка

$$\frac{\rho}{2} < \frac{S_{\text{кон}}}{S_{\text{посл}}} \le \rho, \qquad (9.4)$$

т. е. выигрыш в быстродействии составляет от 0,5ρ до ρ раз.

9.1.3. Конвейеризация путей графа. Условия отображения пути графа алгоритма регламентируются следующими положениями.

УТВЕРЖДЕНИЕ 9.1. Усеченный путь $L_{\text{у}}$ графа алгоритма, для которого не выполняется условие его реализуемости в реальном времени (утверждение 2.2), может быть отображен на конвейерный вычислитель (КВ), если

$$\max_{1 \le j \le J} \{\tau_j\} \le \Delta t(\gamma),$$

где J – количество типов ФУ, соответствующих вершинам пути L_γ, $\Delta t(\gamma) \leftrightarrow v_{i\mu} \in V(\gamma)\ \forall\ v_i \in L_i$.

УТВЕРЖДЕНИЕ 9.2. Для цикла конвейера τ_K справедливо

$$\max_j \{\tau^{(j)}\} \le \tau_K \le \Delta t(\gamma),$$

где $\tau^{(j)}$ – время выполнения операции j-м ФУ, включенным в КВ, $\Delta t(\gamma)$ – шаг дискретизации для вершин конвейеризируемого пути.

Отметим, что для синхронной работы конвейера целесообразно выбирать $\tau_K = \Delta t(\gamma)$.

УТВЕРЖДЕНИЕ 9.3. В ступень конвейера должны входить ФУ, для которых суммарное время выполнения операций меньше или равно его циклу:

$$\sum_{i=1}^{I_\gamma} \tau_i^{(\mu)} \le \tau_K.$$

В целом, считается, что применение конвейерной организации вычислительного процесса позволяет увеличить скорость обработки данных в ρ раз, где ρ – количество ступеней (длина конвейера).

Будем представлять ступени конвейера k-го усеченного пути уровня γ как последовательности вершин, включенных в ступень μ данного пути: $V(L_k(\gamma), \mu)$.

9.2. Пример

Задание. Определить условия отображения усеченных путей ГБС БП, представленного на рис. 7.1 при следующих исходных данных:

– усеченные пути, определенные из (8.1);

– шаги дискретизации в соответствии с табл. 5.1;

– вектор реализации $\boldsymbol{\tau}$ = (1, 1, 1, 1, 1, 1, 1, 1, 15, 12, 2, 11, 1, 16, 1, 1).

Решение. Для определения конвейеризируемых усеченных путей найдем время реализации каждого из них:

$$T_1(1) = \sum_{i \leftrightarrow v \in L_1(1)} \tau(i) = 1 + 2 + 1 + 1 + 1 + 1 = 7 < \Delta t(i);$$

$T_2(1) = 2 < \Delta t(1)$;
$T_3(1) = 2 < \Delta t(1)$;
$T_4(1) = 2 < \Delta t(1)$;
$T_1(2) = 2 < \Delta t(2)$;
$T_2(2) = 32 > \Delta t(2)$;
$T_3(2) = 24 > \Delta t(2)$;
$T_4(2) = 2 < \Delta t(2)$;

$T_1(3) = 1 < \Delta t(3)$;
$T_2(3) = 1 < \Delta t(3)$;
$T_3(3) = 1 < \Delta t(3)$.

Исходя из полученных результатов, делаем вывод об отображении усеченных путей γ-го уровня временной иерархии $L_2(2)$ и $L_3(2)$ на конвейерные вычислители.

Формирование ступеней конвейеров будем проводить исходя из проверки для вершин конвейеризируемых путей условия (9.1). Цикл конвейера равен $\tau_k = \Delta t(2) = 20$.

В первом приближении множество вершин первой ступени пути $L_2(2)$ будет содержать одну начальную вершину этого пути: $V_1(L_2(2), 1) = \{4\}$.

Попытаемся добавить к первой ступени следующую вершину УП, т. е. сформировать первую ступень во втором приближении: $V_2(L_2(2), 1) = \{4, 8\}$.

Время реализации в этом случае будет равно: $T_2(L_2(2), 1) = \tau_4 + \tau_8 = 1 + 15 = 16 < \tau_k$, т. е. включение очередной вершины пути $L_2(2)$ в первую ступень правомерно.

Попытка разместить на первой ступени следующую, 13-ю вершину (в нашем примере единственную оставшуюся), к успеху не приводит, так как суммарное время реализации вершин предполагаемой ступени превышает длительность цикла конвейера: $T_3(L_2(2), 1) = \tau_4 + \tau_8 + \tau_{13} = 16 + 16 = 32 > \tau_k$.

Таким образом, окончательно имеем:
$V(L_2(2), 1) = \{4, 8\}$;
$V(L_2(2), 2) = \{13\}$.

Аналогично для второго конвейеризируемого пути $L_3(2)$:
$V(L_3(2), 1) = \{5, 9\}$;
$V(L_3(2), 2) = \{11\}$.

9.3. Задание

1. Разработать программу определения условий отображения усечённых путей ГВА БП и формирования ступеней конвейера при следующих исходных данных:

– усеченные пути ГБС БП $\{L_v(\gamma),\}$;

– шаги дискретизации $\{\Delta t(\gamma)\}$;

– вектор реализации $\boldsymbol{\tau}$.

2. Проверить работу программы на тестовом примере, заданном преподавателем.

Лабораторная работа № 10
ОПРЕДЕЛЕНИЕ МНОЖЕСТВ СВЕРТЫВАЕМЫХ ВЕРШИН

Целью лабораторной работы является изучение условий возможности свертки вершин.

10.1. Свертка вершин графа

10.1.1. Понятие гомоморфизма. Одними из важнейших унарных операций над графом являются операции гомоморфизма. Пусть дан граф $G = (V, E)$. Выберем во множестве V любые две вершины u, v и построим множество V_1, заменив u, v одной новой вершиной. Построим далее множество ребер E_1. Если ребро в E не инцидентно ни одной из вершин u, v, то это ребро переносится в E_1. Если же ребро в E инцидентно хотя бы одной из вершин u, v, то ребро в E_1 получается из ребра в E заменой вершин новой вершиной. Говорят, что граф $G_1 = (V_1, E_1)$ получается из графа $G = (V, E)$ с помощью операции элементарного гомоморфизма. При изображении графа точками и линиями реализацию этой операции можно трактовать как процесс слияния вершин u, v с непрерывным изменением всех связанных с ними линий. Операции, заключающиеся в последовательном выполнении операций элементарного гомоморфизма, называются *операциями гомоморфизма* или *гомоморфизмом*. Так как получаемые при этих операциях графы содержат меньшее число вершин, чем исходные, то процесс гомоморфного преобразования графов будем называть *сверткой*.

Если после описанной операции исключаются все петли, а все кратные ребра отождествляются, то такая операция называется *простым элементарным гомоморфизмом*. В противном случае она называется *кратным элементарным гомоморфизмом* или, как уже отмечалось, элементарным гомоморфизмом. Операции, заключающиеся в последовательном выполнении простого элементарного гомоморфизма, называются *операциями простого гомоморфизма* или простым гомоморфизмом.

10.1.2. Условие свертки вершин ГБС. С целью дальнейшей минимизации графа и соответствующей ему структуры вычислительной системы целесообразным может быть выполнение над группами (подмножествами) его вершин операции свертки. При этом формируются некоторые множества свертываемых вершин.

ОПРЕДЕЛЕНИЕ 10.1. *Множеством свертываемых вершин* S_i называется множество мощностью больше единицы вершин ГБП или вычислительного графа алгоритма, являющихся прообразами одной из вершин графа вычислительной структуры.

УТВЕРЖДЕНИЕ 10.1. Операция элементарного гомоморфизма может быть выполнена над вершинами u, v графа $G = (V, E)$, если этим вершинам назначены ФУ одного типа при выполнении условия $t_u + \tau_u \le t_v$ на одном цикле обработки данных вершинами u и v, где t_u и t_v – моменты начала выполнения операций, отождествленных с вершинами u и v соответственно; τ_u – время реализации вершины u.

Выполнение операции элементарного гомоморфизма подразумевает реализацию двух или более операций одним и тем же ФУ. Очевидно, что это возможно только в тех случаях, когда соответствующие операции реализуются на взаимно непересекающихся интервалах времени, из чего следует справедливость утверждения.

СЛЕДСТВИЕ 10.1. Свертываемым может быть любое количество вершин, которым назначены ФУ одного типа, принадлежащих одному усеченному пути и не являющихся вершинами разных ступеней конвейера.

С практической точки зрения операция свертки может быть выполнена над некоторыми подмножествами вершин графа, которым назначены ДУ одного типа, выполняющими свои операции на непересекающихся интервалах времени.

В результате реализации процедуры в общем случае формируется некоторое множество подмножеств свертываемых вершин

$$S = \{S^{(1)}, S^{(2)}, \ldots, S^{(M)}\},$$

где $S^{(M)}$, $m = 1, \ldots, M$ – подмножества свертываемых вершин множества S; M – количество подмножеств множества S. Каждое из $S^{(M)}$ имеет вид

$$S^m = (v_1^{(m)}, v_2^{(m)}, \ldots, v_{K_m}^{(m)}),$$

где $K_m = [S_m]$ – мощность подмножества S_m.

10.2. Пример

Задание. Определить множество свертываемых вершин для графа базовой структуры с буферной памятью, полученного в примере 7.3 (см. рис. 7.3). Исходные данные:

– уровни временной иерархии (5.2);

– вектор назначения (6.8); способы отображения путей графа на вычислительную архитектуру, определенные в примере 9.2.

Решение. В соответствии с утверждением 10.1 и следствием 10.1 множество свертываемых вершин является единственным:

$$S_1 = \{7, 14\}. \qquad (10.1)$$

10.3. Задание

1. Разработать программу определения множества свертываемых вершин ГБС БП при следующих исходных данных:

– топология ГБС БП – $G^{\text{БП}}$;

– множество вершин γ-х УВИ $\{\Delta V(\gamma)\}$;

– вектор назначения $\boldsymbol{R}$.

2. Проверить работу программы на тестовом примере, заданном преподавателем.

Лабораторная работа № 11

ОПРЕДЕЛЕНИЕ МНОЖЕСТВА АЛЬТЕРНАТИВНЫХ ВАРИАНТОВ ПРОЕКТИРУЕМОЙ СИСТЕМЫ

Целью лабораторной работы является изучение методологии поиска множества альтернативных вариантов синтезируемой системы.

11.1. Теоретические сведения

В результате выполнения процедуры «назначение функциональных устройств вершинам ГВА» формируется множество векторов назначения $\{\boldsymbol{R}\}$ мощностью W:

$$W=[\boldsymbol{R}]=\prod_{i=1}^{I}\sum_{j=1}^{J} d_{ij}, \qquad (11.1)$$

где I и J – количество вершин и дуг ГВА соответственно; d_{ij} – элементы матрицы соответствия (определение 6.3).

Векторы множества $\{\boldsymbol{R}\}$ для ГВА БП определяют *множество априорных решений* (АР) *1-го уровня*. Каждое из АР 1-го уровня описывается кортежем

$$\begin{aligned}\text{АР1}_w = \{&G_w^{\text{БП}}, \boldsymbol{\varphi}_w^{\text{БП}}, \Delta\boldsymbol{t}, \boldsymbol{R}_w^{\text{БП}}, \boldsymbol{\tau}_w^{\text{БП}},\\ &\{L_y(\gamma)_w^{\text{БП}}\}, \{V(L_k(\gamma), \mu)_w^{\text{БП}}\}\},\end{aligned} \qquad (11.2)$$

где $G_w^{\text{БП}}$ – граф базовой структуры для w-го вектора назначения; $\boldsymbol{\varphi}_w^{\text{БП}}$ – вектор спецификации графа $G_w^{\text{БП}}$; $\Delta\boldsymbol{t}$ – вектор шагов дискретизации; $\boldsymbol{R}_w^{\text{БП}}$ и $\boldsymbol{\tau}_w$ – w-е вектора назначения и реализации соответственно; $\{L_y(\gamma)_w^{\text{БП}}\}$, $w = 1, \ldots, W$ – множество усеченных путей ГБС БП w-го вектора назначения; $\{V(L_k(\gamma), \mu)_w^{\text{БП}}\}$, $w = 1, \ldots, W$ – множество ступеней конвейеров конвейеризируемых усеченных путей ГБС БП w-го вектора назначения; μ – номер ступени конвейера пути $L_k(\gamma)$.

Множества АР 1-го уровня располагаются на 0-м ярусе графа поиска решений (ГПР), представленного на рис. 11.1. Расположение векторов $\boldsymbol{R}_1$–$\boldsymbol{R}_W$ на 0-м ярусе данного графа указывает на тот факт, что каждый из векторов $\boldsymbol{R}_w$ ГБС является порождающим для АР как 1-го, так и более высоких уровней.

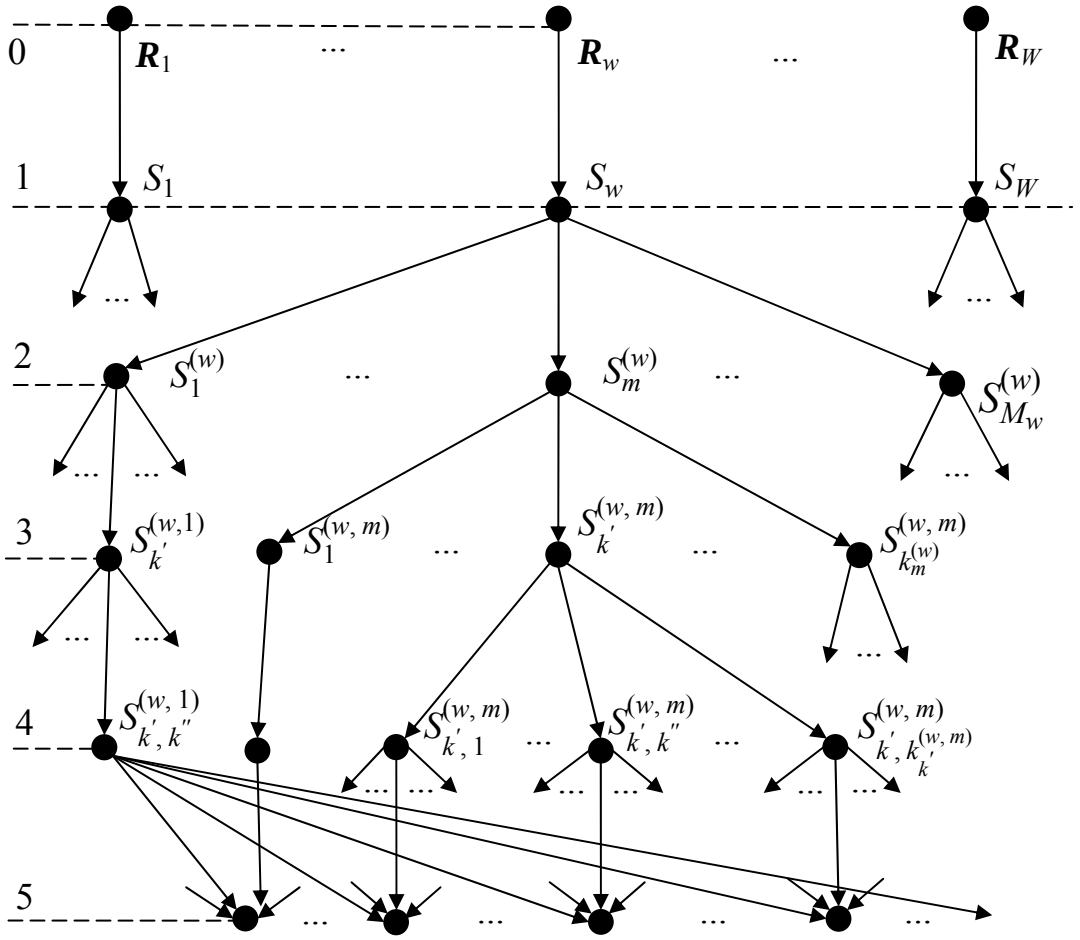

Рис. 11.1. Граф поиска решений

Для каждого вектора $\boldsymbol{R}_w$, $w = 1, \ldots, W$ может быть определено множество S_w подмножеств свертываемых вершин S_w, при этом формируется множество априорных решений 2-го уровня (1-й ярус ГПР на рис. 11.1). АР 2-го уровня описываются кортежем:

$$\text{АР2}_w = \{G_w^{\text{БП}}, \varphi_w^{\text{БП}}, \Delta\boldsymbol{t}, \boldsymbol{R}_w^{\text{БП}}, \boldsymbol{\tau}_w^{\text{БП}}, \{L_y(\gamma)_w^{\text{БП}}\}, \{V(L_k(\gamma), \mu)_w^{\text{БП}}\}, S_w\}, \quad (11.3)$$

где $G_w^{\text{БП}}$, $\varphi_w^{\text{БП}}$, $\Delta\boldsymbol{t}$, $\boldsymbol{R}_w^{\text{БП}}$, $\boldsymbol{\tau}_w^{\text{БП}}$, $\{L_y(\gamma)_w^{\text{БП}}\}$, $\{V(L_k(\gamma), \mu)_w^{\text{БП}}$ имеют тот же смысл, что и в (11.2); S_w – множество свертываемых вершин для вектора назначения $\boldsymbol{R}_w$.

На 2-м ярусе графа поиска решений представлены подмножества $S_m^{(w)}$, $m = 1, \ldots, M_w$ множества S_w:

$$S_m^{(w)} = \left\{S_1^{(w)}, S_2^{(w)}, \ldots, S_m^{(w)}, \ldots, S_{M_w}^{(w)}\right\}, \quad (11.4)$$

здесь $M_w = [S^{(w)}]$ – мощность множества S_w.

Общее количество вершин 2-го яруса ГПР определяется как

$$N_2 = \sum_{w=1}^{W} M_w. \qquad (11.5)$$

Наряду с положительным эффектом, выражающемся в замене каждой из групп $K_w^{(m)}$ однотипных ФУ, реализующих операции, отождествленные с вершинами, входящими в некоторое подмножество $S_m^{(w)}$, одним либо некоторым $K < K_m^{(w)} (K_m^{(w)} = [S_m^{(w)}])$ количеством ФУ, свертка вершин может повлечь негативные последствия, связанные с необходимостью использования мультиплексоров и усложнением управления. В связи с вышесказанным, для дальнейшего анализа всех возможных вариантов системы необходимо учесть все комбинации вершин каждого из множеств $S_m^{(w)}$.

Для этого определим все варианты подмножеств свертываемых вершин для всех $S_m^{(w)}$. При этом количество вершин k, участвующих в свертке, может быть равным:

$k = 0$ (отсутствие свертки),

$k = 2, 3, \ldots, K_w^{(m)}$.

Вариант свертки $K_w^{(m)}$означает свертку всех вершин множества $S_m^{(w)}$; вариант $k = 1$ не рассматривается, так как этот случай определяет $K_w^{(m)}$ однотипных «сверток» каждой из вершин множества вида $S = \{v_i, v_i\}$ («сама с собой»), что эквивалентно варианту $k = 0$.

В силу инвариантности каждого из подмножеств сверток, кроме случаев $k = 0$ и $k = K_m^{(w)}$, назовем соответствующие подмножества $S_{k'}^{(w, m)}$ множества $S_m^{(w)}$ псевдомножествами. Здесь $k' = 1$ при $k = 0$, и $k' = k$ при $k = 2, 3, \ldots, K_m^{(w)}$.

Множество псевдомножеств множества $S_m^{(w)}$ представлено на третьем ярусе ГПР (рис. 11.1). Мощности дочерних псевдомножеств для каждого из множеств $S_m^{(w)}$ равны:

$$\left[S_1^{(w, m)}\right] = \emptyset, \ \left[S_2^{(w, m)}\right] = 2, \ \left[S_3^{(w, m)}\right] = 3, \ \ldots, \ \left[S_{K_m^{(w)}}^{(w, m)}\right] = K_m^{(w)}. \qquad (11.6)$$

Количество вершин данного яруса:

$$N_3 = \sum_{w=1}^{W} M_w \sum_{m=1}^{M_w} K_m^{(w)}. \qquad (11.7)$$

Каждое псевдомножество $S_{k'}^{(w, m)}$, $k' = 2, 3, \ldots, K_m^{(w)}$ может быть представлено множеством комбинаций из k' элементов, принадлежащих множеству S_w:

$$S_{k'}^{(w,\,m)}=\left\{S_{k',\,1}^{(w,\,m)},\ S_{k',\,2}^{(w,\,m)},\ \ldots,\ S_{k',\,k''}^{(w,\,m)},\ \ldots,\ S_{k',\,K_{k'}^{(w,\,m)}}^{(w,\,m)}\right\},\qquad(11.8)$$

где

$$K_{k'}^{(w,\,m)}=C_{K_m^{(w)}}^{k'}=C_{K_m^{(w)}}^{k},\qquad(11.9)$$

где $C_{K_m^{(w)}}^{k'}$ – количество сочетаний из $K_m^{(w)}$ по k:

$$C_{K_m^{(w)}}^{k}=K_m^{(w)}!/(k!\left(K_m^{(w)}-k\right)!),\ k=2,\ 3,\ \ldots,\ K_m^{(w)}.\qquad(11.10)$$

Таким образом, каждое множество $S_m^{(w)}$ порождает $N_m^{(w)}$ дочерних подмножеств

$$N_m^{(w)}=\sum_{k=2}^{K_m^{(w)}}K_m^{(w)}!\,/\left(k!\left(K_m^{(w)}-k\right)!\right)+1.\qquad(11.11)$$

Первое слагаемое в (11.11) представляет собой сумму количеств сочетаний по k из $K_m^{(w)}$ для $k = 2, \ldots, K_m^{(w)}$, второе слагаемое (+1) указывает на вариант $S_1^{(w,\,m)}=\emptyset$ (отсутствие свертки для вершин множества $S_m^{(w)}$).

Дочернее подмножество псевдомножества $S_k^{(w,\,m)}$ изображено на четвертом ярусе ГПР (рис. 11.1).

Отметим, что результат, эквивалентный (11.11), может быть получен с использованием степенных множеств.

Каждое из множеств $S=S_m^{(k)}$ может быть представлено степенным множеством PS (powerset) его подмножеств

$$S\leftrightarrow PS=2^S,$$

включающим все возможные подмножества, в том числе пустое подмножество и само множество S.

Мощность степенного множества PS равна

$$[PS]=2^{[S]}.$$

Исключив из степенного множества вида (11.12) подмножества мощностью 1, получим

$$PS'=PS-\bigcup_{l=1}^{L}S_l,\qquad(11.12)$$

где PS' – модифицированное степенное множество множества S; S_l – l-е одноэлементное подмножество множества S; $L = [S]$ – количество элементов множества S.

Мощность модифицированного степенного множества (11.12) определяется из (11.13):

$$[PS'] = 2^{[S]} - [S]. \qquad (11.13)$$

Таким образом, для вычисления количества дочерних множеств $N_m^{(w)}$ для $S_m^{(w)}$ на четвертом ярусе ГПР можно использовать выражение

$$N_m^{(w)} = 2^{[S_m^{(w)}]} - [S_m^{(w)}]. \qquad (11.14)$$

Окончательно априорные решения, которые могут быть получены в результате перебора всех вариантов сверток для каждого из векторов $\boldsymbol{R}_w$, формируются как декартовы произведения M_w подмножеств четвертого яруса:

$$S_w = S_1^{(w)} \times S_2^{(w)} \times \ldots \times S_{M_w}^{(w)} = \left\{ \left(s_1^{(w)}, s_2^{(w)}, \ldots, s_{M_w}^{(w)} \right) : s_m^{(w)} \in S_m^{(w)}, \atop m = 1, 2, \ldots, M_w \right\}. \qquad (11.15)$$

Здесь каждое множество $S_m^{(w)}$ представлено своими подмножествами четвертого яруса:

$$S_m^{(w)} = \left\{ S_1^{(w,\,m)}, \ldots, S_{k'}^{(w,\,m)}, \ldots, S_{K_m^{(w)}}^{(w,\,m)} \right\}, \qquad (11.16)$$

где $S_{k'}^{(w,\,m)}$ – множество, определяющее свертки k' вершин:

$$S_{k'}^{(w)} = \left\{ S_{k',\,1}^{(w,\,m)}, \ldots, S_{k',\,k''}^{(w,\,m)}, \ldots, S_{k',\,K_m^{(w)}}^{(w,\,m)} \right\}. \qquad (11.17)$$

Здесь $S_{k',\,k''}^{(w,\,m)}$, $k'' = 1, \ldots, K_{k'}$ – конкретная реализация псевдомножества $S_{k'}^{(w,\,m)}$, $K_{k'}^{(w,\,m)}$ определено выражением (11.9).

Тогда количество N_w априорных вариантов для одного вектора $\boldsymbol{R}_w$ (5-й ярус ГПР) определяется как произведение мощностей множеств $S_m^{(w)}$:

$$N_w = \left[S_1^{(w)} \right] \cdot \left[S_2^{(w)} \right] \cdot \ldots \cdot \left[S_{M_w}^{(w)} \right] = \prod_{m=1}^{M_w} S_m^{(w)}.$$

Используя для расчета выражение (12.11), получим

$$N_w = \prod_{m=1}^{M_w} \left(\sum_{k=2}^{K_m^{(w)}} K_m^{(w)}! \Big/ \left(k! \left(K_m^{(w)} - k \right)! \right) + 1 \right).$$

В этом случае общее число вариантов для всех векторов назначения будет вычисляться по выражению

$$N = \sum_{w=1}^{W} \prod_{m=1}^{M_w} \left(\sum_{k=2}^{K_m^{(w)}} K_m^{(w)}! \Big/ \left(k! \left(\boldsymbol{K}_{\boldsymbol{m}}^{(w)} - \boldsymbol{k} \right)! \right) + 1 \right). \qquad (11.18)$$

При подсчете $N_m^{(w)}$ по выражению (12.14), получим

$$N = \sum_{w=1}^{W} \prod_{m=1}^{M_w} \left(2^{\left[S_m^{(w)} \right]} - \left[S_m^{(w)} \right] \right). \qquad (11.19)$$

Будем называть решения, полученные на пятом ярусе ГПР, априорными решениями АР3 третьего уровня. Каждое из таких решений описывается кортежем:

$$\text{АР3}_w = \{ G_w^{\text{БП}}, \boldsymbol{\varphi}_w^{\text{БП}}, \Delta \boldsymbol{t}, \boldsymbol{R}_w^{\text{БП}}, \boldsymbol{\tau}_w^{\text{БП}}, \{ L_\text{у}(\gamma)_w^{\text{БП}} \}, \{ V(L_k(\gamma), \mu)_w^{\text{БП}} \}, S_n^{(w)} \},$$

где $S_n^{(w)}$, $n = 1, \ldots, K_w$ – n-е подмножество $S_n^{(w)}$, формируемое из (11.15).

11.2. Пример

Определить множество априорных решений для системы, реализующей граф G со спецификацией вершин $\boldsymbol{\varphi}$ для векторов назначения $\boldsymbol{R}$ и $\boldsymbol{\tau}$ соответственно и множеством подмножеств свертываемых вершин

$$S = \{ S_1, S_2, S_3 \},$$

где $S_1 = \{5, 8\}$, $S_2 = \{3, 9, 12\}$, $S_3 = \{10, 14\}$.

Определим количество вариантов, используя (11.12):

$$N_1 = 2^{[S_1]} - [S_1] = 4 - 2 = 2;$$
$$N_2 = 2^{[S_2]} - [S_2] = 8 - 3 = 5;$$
$$N_3 = 2^{[S_3]} - [S_3] = 4 - 2 = 2.$$

Количество вариантов будет равно:

$$N = N_1 \cdot N_2 \cdot N_3.$$

Для определения вариантов проектных решений сформируем подмножества модифицированных степенных множеств для S.

$PS_1' = \{\emptyset, (5, 8)\}$;

$PS_2' = \{\emptyset, (3, 9), (9, 12), (3, 12), (3, 9, 12)\}$;

$PS_3' = \{\emptyset, (10, 14)\}$.

Тогда искомые варианты решений будут характеризоваться одинаковыми наборами элементов G, $\boldsymbol{r}$, $\boldsymbol{R}$, $\boldsymbol{\tau}$ и множествами S_k, $k = 1, \ldots, 20$.

$S_1 = \{\emptyset, \emptyset, \emptyset\}$;

$S_2 = \{\emptyset, \emptyset, (10, 14)\}$;

$S_3 = \{\emptyset, (3, 9), \emptyset\}$;

$S_4 = \{\emptyset, (3, 9), (10, 14)\}$;

$S_5 = \{\emptyset, (9, 12), \emptyset\}$;

$S_6 = \{\emptyset, (9, 12), (10, 14)\}$;

…

$S_{20} = \{(5, 8), (3, 9, 12), (10, 14)\}$.

11.3. Задание

1. Разработать программу, позволяющую:

а) рассчитать количество альтернативных вариантов проектируемой системы;

б) определить множество априорных решений 2-го уровня.

Исходные данные:

– граф базовой структуры с буферной памятью $G^{\text{БС}}$;

– спецификация вершин ГБС БП $\boldsymbol{\varphi}$;

– вектор назначения $\boldsymbol{R}$;

– вектор реализации $\boldsymbol{\tau}$;

– множество подмножеств свертываемых вершин $S = \{S_1, S_2, ..., S_M\}$.

2. Проверить работу программы на тестовом примере.

Лабораторная работа № 12
ПОСТРОЕНИЕ ВЫЧИСЛИТЕЛЬНОГО ГРАФА АЛГОРИТМА

Целью лабораторной работы является изучение принципов формирования вычислительного графа алгоритма.

12.1. Определение и принципы формирования вычислительного графа алгоритма

12.1.1. Служебные операции и служебные функциональные устройства. Для того чтобы заставить работать ФУ, необходимо на их входы подать соответствующие данные. Но даже передача данных одного ФУ к другому требует временных затрат. Если же по каким-либо причинам нам приходится для реализации алгоритма пользоваться одним из видов памяти системы, то дополнительные временные затраты на работу с памятью могут оказаться значительными. Более того, именно каналы связи с памятью часто оказываются одним из самых узких мест вычислительной системы.

Все ФУ любой конкретной системы можно разделить на две группы: ФУ, выполняющие основные операции алгоритма, и ФУ, обеспечивающие их эффективную работу. Как было отмечено ранее (см. л. р. № 1), к первой группе относятся ФУ, непосредственно выполняющие операции алгоритма. Ко второй группе можно отнести устройства ввода-вывода, каналы передачи данных, каналы связи с различными видами памяти и многое другое, в частности мультиплексоры.

Временные затраты на работу с памятью учтены при построении графа базовой структуры с буферной памятью. Вершинам, отражающим эти задержки, назначены ФУ типа «память», в простейшем случае – регистры. Другие служебные операции реализуются служебными функциональными устройствами.

Тогда реализацию любого алгоритма на конкретной вычислительной системе можно рассматривать как реализацию некоторого расширенного алгоритма, полученного из исходного добавлением операций, не меняющих каких-либо данных, но требующих для своего выполнения определенного времени.

Граф расширенного алгоритма получается из графа исходного алгоритма путем последовательного применения операций «добавление вершины», соответствующих использованию служебных ФУ вычислительной системы. Для расширенного алгоритма справедливы все полученные ранее результаты. В частности, по нему можно оценивать реальное время реализации алгоритма, а затем исследовать эффективность использования ФУ, основных или служебных, всех или только части их так, как это диктуется необходимостью. Связь графа расширенного алгоритма с вычислительной системой отражена в его названии – *вычислительный* граф алгоритма.

12.1.2. Теоретические аспекты построения вычислительного графа алгоритма. Построение ВГА базируется на следующих положениях.

ОПРЕДЕЛЕНИЕ 12.1. Граф расширенного алгоритма, формируемый из графа исходного алгоритма путем последовательного применения операции добавления вершины, называется вычислительным графом алгоритма.

УТВЕРЖДЕНИЕ 12.1. Операция добавления вершины должна выполняться для дуг, входных по отношению к вершинам, включенным в одно из множеств свертываемых вершин, если начальным вершинам этих дуг не назначены ФУ с магистральным выходом.

УТВЕРЖДЕНИЕ 12.2. Операция добавления вершины должна быть выполнена для дуги, инцидентной вершине, которой назначен блок памяти, если этот блок не работает в режиме постоянного чтения.

12.1.3. Мультиплексирование потоков данных. Одной их типичных ситуаций введения в систему служебных ФУ является свертка вершин. Предположим, что для некоторого графа сформировано множество свертываемых вершин $S = \{k, n\}$. Подграф G', содержащий указанные вершины, изображен на рис. 12.1.

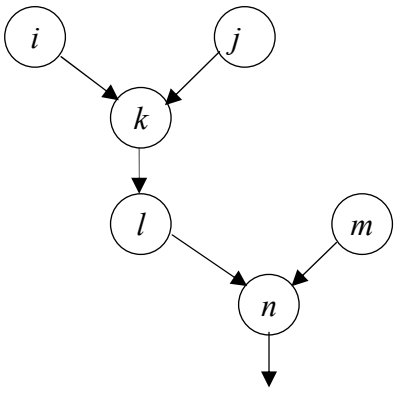

Рис. 12.1. Подграф, содержащий свертываемые вершины *k* и *n*

На одном из последних этапов синтеза (процедура построения графа вычислительной структуры) прообразы k и n будут заменены образом p. Т. е. подграф G' будет преобразован в подграф, изображенный на рис. 12.2.

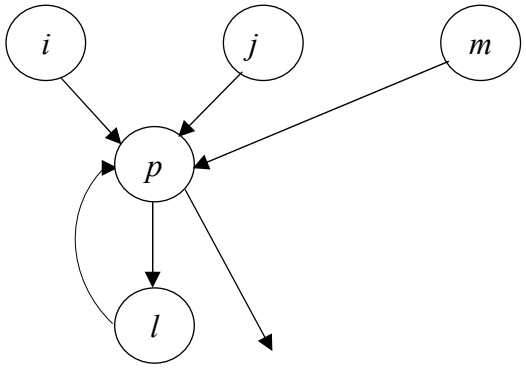

Рис. 12.2. Подграф, полученный
в результате свертки вершин k и n:
p – образ вершин k и n

Тогда в вычислительной системе на каждый из входов ФУ, назначенного вершине p (ФУ$_p$), в различные моменты времени будут поступать данные от различных ФУ: ФУ$_i$ и ФУ$_j$, назначенных вершинам V_i и V_j соответственно – в момент времени t_1, и ФУ$_l$, ФУ$_m$, назначенных вершинам с соответствующими индексами – в момент $t_2 > t_1$. Для разделения источников информации для ФУ$_p$ необходимо использовать служебные устройства типа «мультиплексор», как показано на рис. 12.3.

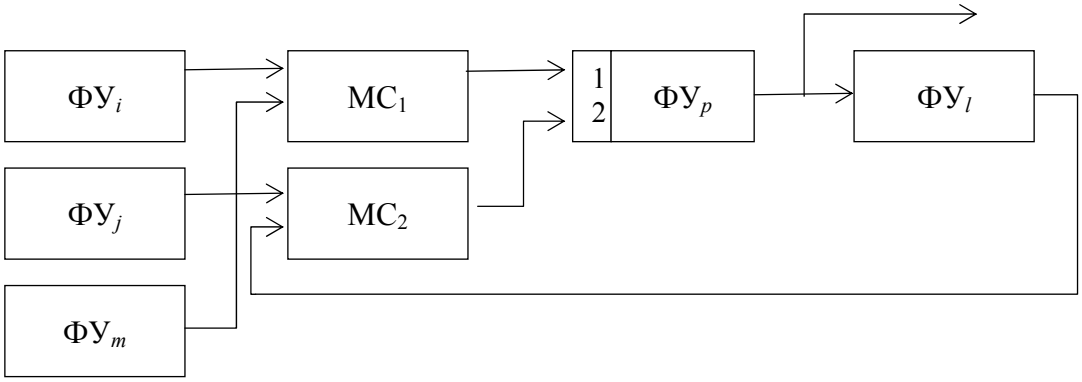

Рис. 12.3. Схема реализации подграфа G':
МС$_1$, МС$_2$ – мультиплексоры

Учитывая повторное прохождение данных через мультиплексоры на одном цикле работы системы (количество повторений Q равно мощности множества свертываемых вершин – в нашем

примере – два), операцию добавления вершины при формировании вычислительного графа алгоритма необходимо выполнять для всех дуг, входящих в каждую из вершин множества S.

Подграф $G'^{\text{ВГА}}$ вычислительного графа алгоритма, сформированный путем преобразования подграфа G', изображен на рис. 12.4.

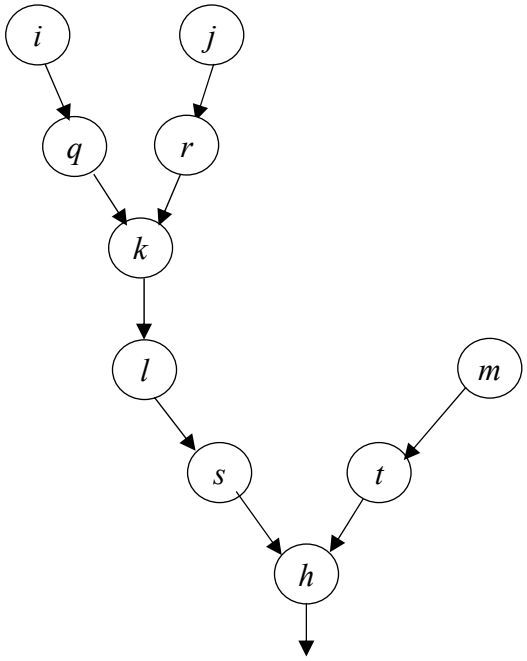

Рис. 12.4. Подграф $G'^{\text{ВГА}}$ вычислительного графа алгоритма

12.2. Пример

Для графа базовой структуры с буферной памятью, представленного на рис. 7.3, при множестве свертываемых вершин (10.1) вычислительный граф алгоритма будет иметь вид, представленный на рис. 12.1. Спецификация вершин данного графа формируется из спецификаций вершин ГБС БП путем добавления к ней элементов с индексами добавленных вершин, которым в соответствие ставится операция мультиплексирования.

Вектор назначения для данного графа:

$$\boldsymbol{R}^{\text{ВГА}} = (19, 19, 19, 19, 19, 19, 21, 5, 10, 8,$$
$$7, 11, 21, 20, 13, 13, 13, 13, 13, 16, 16, 16, 16), \qquad (12.1)$$

соответствующий данному вектору назначения вектор реализации:

$$\boldsymbol{\tau}^{\text{ВГА}} = (\text{l}, \text{l}, \text{l}, \text{l}, \text{l}, \text{l}, \text{l}, 1, 15, 12, 2, \text{l},$$
$$\text{l}, \text{l}, 16, 1, 1, 2, 2, 2, 2, 2, 1, 1, 1, 1). \qquad (12.2)$$

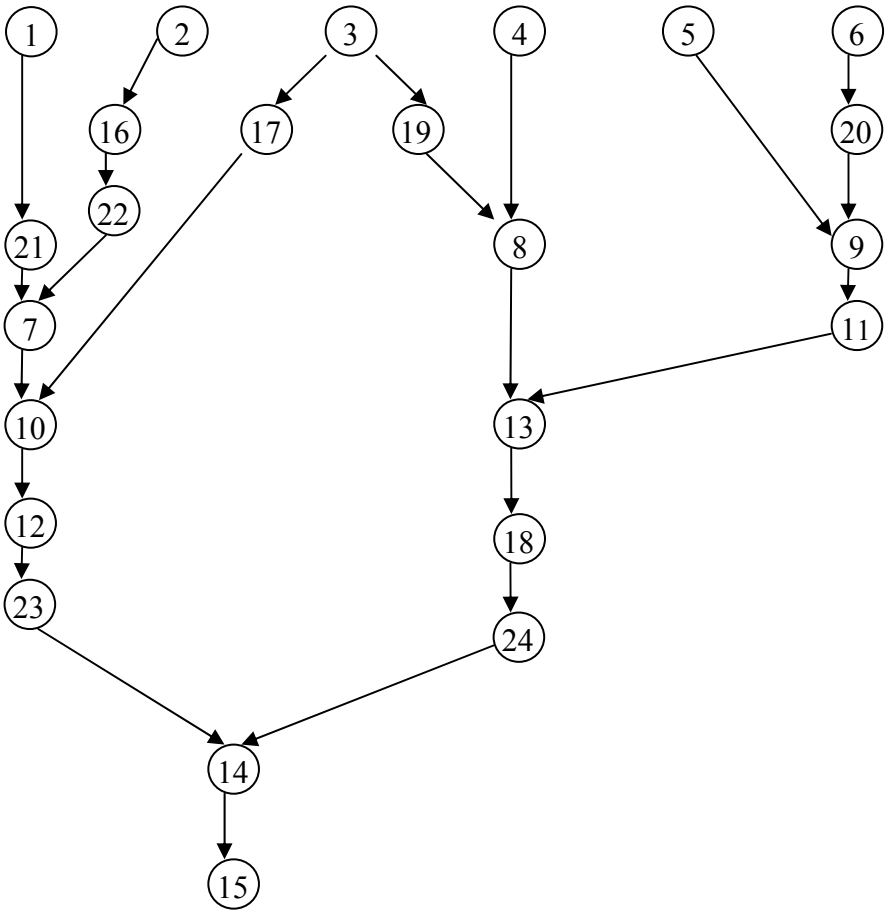

Рис. 12.5. Вычислительный граф алгоритма

12.3. Задание

1. Разработать программу формирования ВГА, используя следующие исходные данные:

– топология ГБС БП;

– множество подмножеств свертываемых вершин:

$$S = \{S^{(1)}, S^{(2)}, \ldots, S^{(M)}\},$$

где $S^{(m)}$, $m = 1, \ldots, M$ – подмножества свертываемых вершин множества S; M – количество подмножеств множества S.

2. Продемонстрировать работу созданного программного продукта на тестовом примере, заданном преподавателем.

Лабораторная работа № 13

ПЕРВАЯ ПРОВЕРКА РЕАЛИЗУЕМОСТИ ВЫЧИСЛИТЕЛЬНОЙ СИСТЕМЫ РЕАЛЬНОГО ВРЕМЕНИ

Целью лабораторной работы является изучение условия реализуемости ВСРВ.

13.1. Условие реализуемости графа в реальном времени

Добавление вершин в процессе преобразования исходного графа приведет к увеличению времени реализации путей графа, что может нарушить условие реализуемости пути в реальном времени. Таким образом, во избежание синтеза системы, заведомо не удовлетворяющей требованиям реального времени, необходимо на раннем этапе синтеза провести анализ реализуемости системы на выбранном векторе назначения. Условие реализуемости вычислительной системы регламентируется следующим положением.

УТВЕРЖДЕНИЕ 13.1. Необходимым условием реализуемости вычислительной структуры в реальном времени является выполнение для всех неконвейеризируемых путей вычислительного графа алгоритма условия

$$\sum_{i \in L_{\text{у}}} \tau(i) \le \Delta t(\gamma), \qquad (13.1)$$

где $L_{\text{у}}$ – усеченный путь уровня γ.

Справедливость утверждения следует из условия реализуемости пути в реальном времени (утверждение 2.2).

СЛЕДСТВИЕ 13.1. Если суммарное время реализации вершин неконвейеризируемого пути вычислительного графа алгоритма, содержащего вершины из множества свертываемых вершин, превышает шаг дискретизации соответствующего УВИ, вычислительная структура реального времени при выбранном векторе назначения является нереализуемой. В случае невыполнения условия (13.1) хотя бы для одного из усеченных путей необходимо

сформировать сообщение о нереализуемости ВСРВ на выбранном векторе назначения и перейти к обработке очередного вектора из множества $\{\boldsymbol{R}\}$, определенных на четвертом шаге проектирования (см. л. р. № 6).

13.2. Пример

Проверим на реализуемость в реальном времени вычислительную структуру, описываемую вектором реализации (6.2) при уровнях временной иерархии (5.2) и шагах дискретизации (5.1), приведенными ниже усеченными путями вычислительного графа алгоритма.

$L^{\text{ВГА}}{}_1(1) = (1, 21, 7, 10, 12, 23, 14, 15)$;

$L^{\text{ВГА}}{}_2(1) = (16, 22)$;

$L^{\text{ВГА}}{}_3(1) = (17)$;

$L^{\text{ВГА}}{}_4(1) = (18, 24)$;

$L^{\text{ВГА}}{}_1(2) = (2)$;

$L^{\text{ВГА}}{}_2(2) = (4, 8, 13)$;

$L^{\text{ВГА}}{}_3(2) = (5, 9, 11)$;

$L^{\text{ВГА}}{}_4(2) = (19)$;

$L^{\text{ВГА}}{}_5(2) = (20)$;

$L^{\text{ВГА}}{}_1(3) = (3)$;

$L^{\text{ВГА}}{}_2(3) = (3)$;

$L^{\text{ВГА}}{}_3(3) = (6)$.

Обратим внимание на то, что вершина v_3 включена в два усеченных пути как точка ветвления. Время реализации усеченных неконвейеризируемых путей:

$T_1(1) = 9 < \Delta t(1) = 10$;

$T_2(1) = 3 < \Delta t(1)$;

$T_3(1) = 2 < \Delta t(1)$;

$T_4(1) = 3 < \Delta t(1)$;

$T_1(2) = 1 < \Delta t(1)$;

$T_4(2) = 2 < \Delta t(2) = 20$;

$T_5(2) = 2 < \Delta t(2)$;

$T_1(3) = T_2(3) = 1 < \Delta t(3) = 40;$

$T_3(3) = 1 < \Delta t(3).$

Из полученных результатов делаем вывод о выполнении первого необходимого условия реализуемости ВСРВ.

13.3. Задание

1. Разработать программу анализа реализуемости ВСРВ на выбранном векторе назначения. Исходные данные:

– векторы усеченных путей ВГА $\{\boldsymbol{L}_k(\gamma)\}$;

– вектор шагов дискретизации Δ;

– вектор реализации $\boldsymbol{\tau}$.

2. Проверить работу программы на заданном примере.

Лабораторная работа № 14
ФОРМИРОВАНИЕ ВЕКТОРА ВРЕМЕННОЙ РАЗВЕРТКИ

Целью лабораторной работы является изучение принципов синхронизации вычислительной системы реального времени.

14.1. Методика расчета координат вектора временной развертки

Программа управления ВСРВ задается вектором *временной развертки* (ВВР).

ОПРЕДЕЛЕНИЕ 14.1. *Вектором временной развертки базовой вычислительной структуры* называется вектор-столбец $\boldsymbol{t} = (t_1, t_2, \ldots, t_n)$, каждая из координат $\boldsymbol{t}_i$, $i = 1, 2, \ldots, I$ которого является моментом включения ФУ, соответствующего i-й вершине.

Принцип формирования координат ВВР определяется способом отображения конкретных путей ВГА (вычислительного графа алгоритма) на вычислительную архитектуру.

ОПРЕДЕЛЕНИЕ 14.2. Нулевой координатой вектора временной развертки будем называть координату t_i^0, соответствующую первому включению ФУ, назначенного начальной вершине базисного пути. Присвоим этой координате нулевое значение.

УТВЕРЖДЕНИЕ 14.1. Элементы $t(i)$ вектора временной развертки усеченного пути L при известном значении координаты ВВР для начальной вершины $t(1)$ бесконвейерного пути определяются выражением

$$t(i)^{(L)} = t(1)^{(L)} + \sum_{j=1}^{i\text{-}1} \tau(j)^{(L)}, \qquad (14.1)$$

где $t(1)^{(L)}$ – элемент ВВР для начальной вершины пути L, $\tau(j)$, $j = 1, \ldots, i - 1$ – элементы вектора реализации, соответствующие последовательным вершинам пути L.

Справедливость утверждения следует из требования выполнения условий реализуемости вычислительной структуры в реальном времени – утверждение 13.1 (см. рис. 14.1).

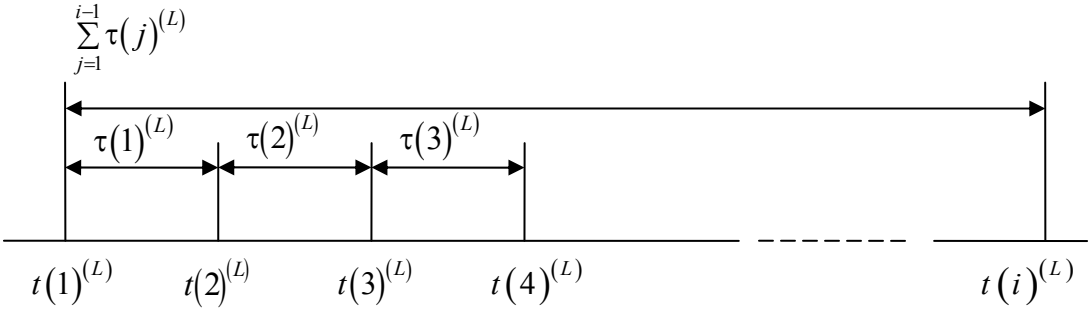

Рис. 14.1. Формирование *i*-й координаты вектора временной развертки при известной координате $t(1)^{(L)}$: $t(1)^{(L)}$, $t(2)^{(L)}$, $t(3)^{(L)}$ – координаты вектора временной развертки пути *L*; $\tau(1)^{(L)}$, $\tau(2)^{(L)}$, $\tau(3)^{(L)}$ – координаты вектора реализации

ПРИМЕЧАНИЕ 14.1. С учетом возможности сокращения первой ступени при небольшом значении ρ целесообразно пользоваться формулой

$$t(i_\tau)^{(L)} = t(1)^{(L)} + \sum_{k=1}^{K_1} \tau(k)^{(1,\,L)} + (\mu - 2)\tau_K + \sum_{j=1}^{i_r - 1} \tau(j)^{(\mu,\,L)}, \quad (14.2)$$

где K_1 – количество вершин первой ступени; $\tau(k)^{(1,L)}$ – время реализации *k*-й вершины 1-й ступени *L*-го пути.

УТВЕРЖДЕНИЕ 14.2. Значение элемента $t(i_\mu)^{(L)}\ \forall\ i = \overline{1, J_\mu^{(L)}}$, $\mu = \overline{1, p}$ вектора временной развертки конвейеризируемого пути при известной координате для начальной вершины этого пути определяется выражением

$$t_w = t\left(i_\mu\right)^{(L)} = t(1)^{(L)} + (\mu - 1)\tau_K + \sum_{j=1}^{\mu - 1} \tau(j)^{(\mu,\,L)}, \quad (14.3)$$

где $t\left(i_\mu\right)^{(L)}$ – координата вектора временной развёртки *i*-й вершины μ-й ступени пути *L*.

Справедливость утверждения следует из утверждения 13.1. Случай проиллюстрирован на рис. 14.2.

УТВЕРЖДЕНИЕ 14.3. Координата вектора временной развертки для начальной вершины конвейеризируемого пути *L*, дуга из конечной вершины которого входит в вершину *w* с известной координатой ВВР *t*(*w*), определяется соотношением

$$t(1)^{(L)} = t(w) - \tau_K \cdot \rho^{(L)}, \quad (14.4)$$

где $\rho^{(L)}$ – число ступеней конвейера усеченного пути *L*.

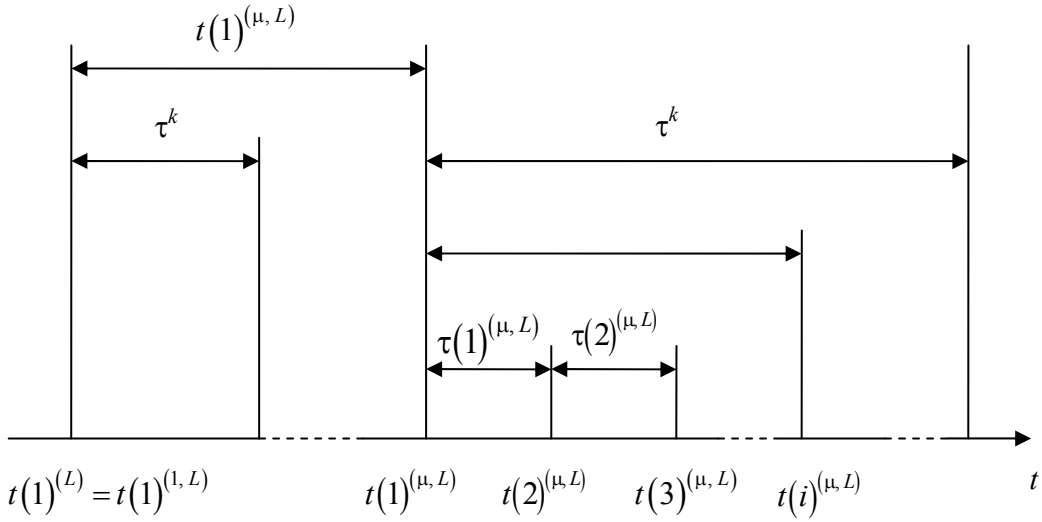

Рис. 14.2. Формирование координаты вектора
временной развертки конвейеризируемого пути L:
$t(1)^{(1, L)}$, $t(1)^{(\mu, L)}$ – координаты ВВР начальных вершин 1-й, ..., μ-й ступеней;
$\tau(1)^{(\mu, L)}$, $\tau(2)^{(L)}$ – координаты вектора реализации

Для конвейеризируемого пути данные на выходе ρ-й ступени конвейера формируются через интервал времени, равный $\tau_K \cdot \rho^{(L)}$ после включения первого ФУ первой ступени конвейера, тогда момент готовности этих данных определяется из выражения

$$t_{\text{вых}} = t(1)^{(L)} - \tau_K \cdot \rho^{(L)}. \qquad (14.5)$$

Эти данные должны быть использованы вершиной w в момент $t(w)$, и из условия реализуемости путей L в реальном времени потребуем $t_{\text{вых}} = t(w)$. Отсюда следует справедливость утверждения.

ПРИМЕЧАНИЕ 14.2. При малых значениях ρ целесообразно использовать более точную формулу

$$t(1)^{(L)} = t(w) - \sum_{k=1}^{K_i} \tau(k)^{(1, L)} - \sum_{j=1}^{J_\rho} \tau(j)^{(\rho, L)} - (\rho - 1)\tau_K, \qquad (14.6)$$

где K_i и $\tau(k)^{(1, L)}$ имеют тот же смысл, что и в (14.2); J_ρ – количество вершин ρ-й ступени; $\tau(j)^{(\rho, L)}$ – время выполнения операции, отождествленной с j-й вершиной ρ-й ступени пути L. Справедливость примечаний 14.1 и 14.2 следует из утверждения 13.1.

УТВЕРЖДЕНИЕ 14.4. Значение элемента вектора временной развертки $t(i) \forall i = \overline{1, k}$ неконвейеризируемого пути, из конечной вершины которого идет дуга в вершину w с известной координатой вектора временной развертки $t(w)$, определяется из выражения

$$t(i)^{(L)} = t(w) - \sum_{l=0}^{k-i} \tau(k-l)^{(L)}. \qquad (14.7)$$

Иллюстрация случая приведена на рис. 14.3.

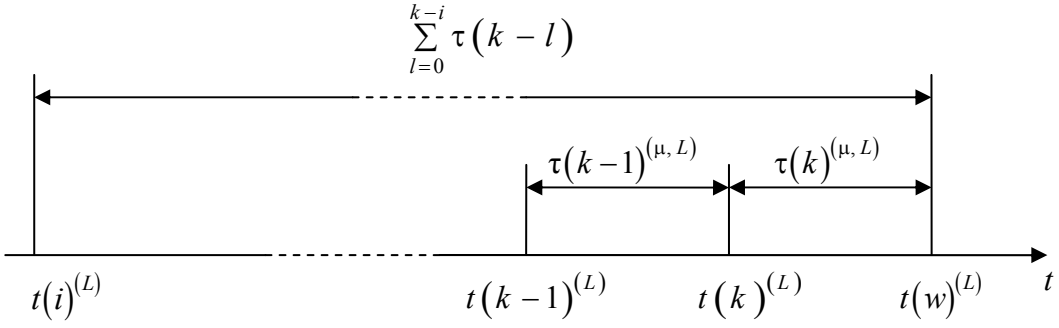

Рис. 14.3. Формирование *i*-й координаты вектора временной развертки при известной координате $t(w)$ вершины, в которую входит дуга из конечной вершины пути

УТВЕРЖДЕНИЕ 14.5. Координата вектора временной развертки вершины пути, выходящего из вершины с известной координатой вектора временной развертки $t(w)$, определяется из выражения

$$t(i)^{(L)} = t(w) - \sum_{j=1}^{i-1} \tau(j). \qquad (14.8)$$

Заметим, что данный случай характерен для циклических алгоритмов.

14.2. Пример

Задание. Сформировать вектор временной развертки для вычислительного графа алгоритма, изображенного на рис. 12.5, при следующих исходных данных:

– вектор назначения (12.1);

– вектор реализации (12.2);

– вектор шагов дискретизации (5.4);

– конвейеризируемые пути из примера 9.2;

– усеченные пути ВГА из примера 13.2.

Решение. За нулевую координату ВВР выберем координату, соответствующую начальной вершине базисного пути:

$t_1 = t_1(1) = 0$.

Для вычисления элементов, соответствующих последующим вершинам данного пути, используем выражение (14.1).

$t_{21} = t_1 + \tau_1 = 0 + 1 = 1;$

$t_7 = t_1 + (\tau_1 + \tau_{21}) = t_{21} + \tau_{21} = 1 + 1 = 2;$

$t_{10} = t_7 + \tau_7 = 2 + 1 = 3.$

В результате аналогичных вычислений получим:

$t_{10} = 3;$

$t_{12} = 4;$

$t_{23} = 5;$

$t_{14} = 6;$

$t_{15} = 7.$

Элементы ВВР для вершин пути $L_2(1)$, входящего в вершину v_7 с известной координатой ВВР $t_7 = 2$, определим из (14.4):

$t_{22} = t_7 + \tau_{22} = 2 - 1 = 1;$

$t_{16} = t_7 + (\tau_{22} + \tau_{16}) = t_{22} + \tau_{16} = 1 - 2 = -1.$

Аналогично вычисляются значения координат ВВР для вершин, принадлежащих остальным неконвейеризируемым путям.

Для определения координат ВВР для начальной вершины конвейеризируемого пути $L_2(2)$ воспользуемся выражением (14.3):

$t_4 = t_{14} - \Delta t(2) \cdot \rho_2(2) = 24 - 20 \cdot 2 = -34.$

Из (14.1) имеем

$t_8 = t_4 + \tau_4 = -33.$

Из (14.2) получим

$t_{13} = t_4 + \rho = -34 + 20 = -14.$

Для вершин пути $L_3(2)$ расчет проводится аналогично.

Вектора временной развертки $\boldsymbol{t}_i(\gamma)$ усеченных путей вычислительного графа алгоритма в соответствии с методикой, рассмотренной в п. 14.1 будут следующими:

$\boldsymbol{t}_1(1) = (0, 1, 2, 3, 4, 5, 6, 7, 8);$

$\boldsymbol{t}_2(1) = (-2, 1);$

$\boldsymbol{t}_3(1) = (2);$

$\boldsymbol{t}_4(1) = (4, 6);$

$\boldsymbol{t}_1(2) = (-2);$

$\boldsymbol{t}_2(2) = (-34, -33, -14);$

$\boldsymbol{t}_3(2) = (-56, -55, -36);$

$\boldsymbol{t}_4(2) = (-37);$

$\boldsymbol{t}_1(3) = (-57);$

$\boldsymbol{t}_2(3) = (1)$;

$\boldsymbol{t}_3(3) = (-38)$;

$\boldsymbol{t}_4(3) = (-58)$.

Отметим, что при определении координат векторов временной развертки конвейеризируемых путей использованы основные утверждения (13.1, 14.1–14.6) без учета примечаний 14.1 и 14.2.

Значения координаты ВВР $\boldsymbol{t}_3$ из $\boldsymbol{t}_1(3)$ и $\boldsymbol{t}_3(3)$ принимаем равной 36. Вектор временной развертки формируется из векторов временной развертки усеченных путей (рис. 14.4).

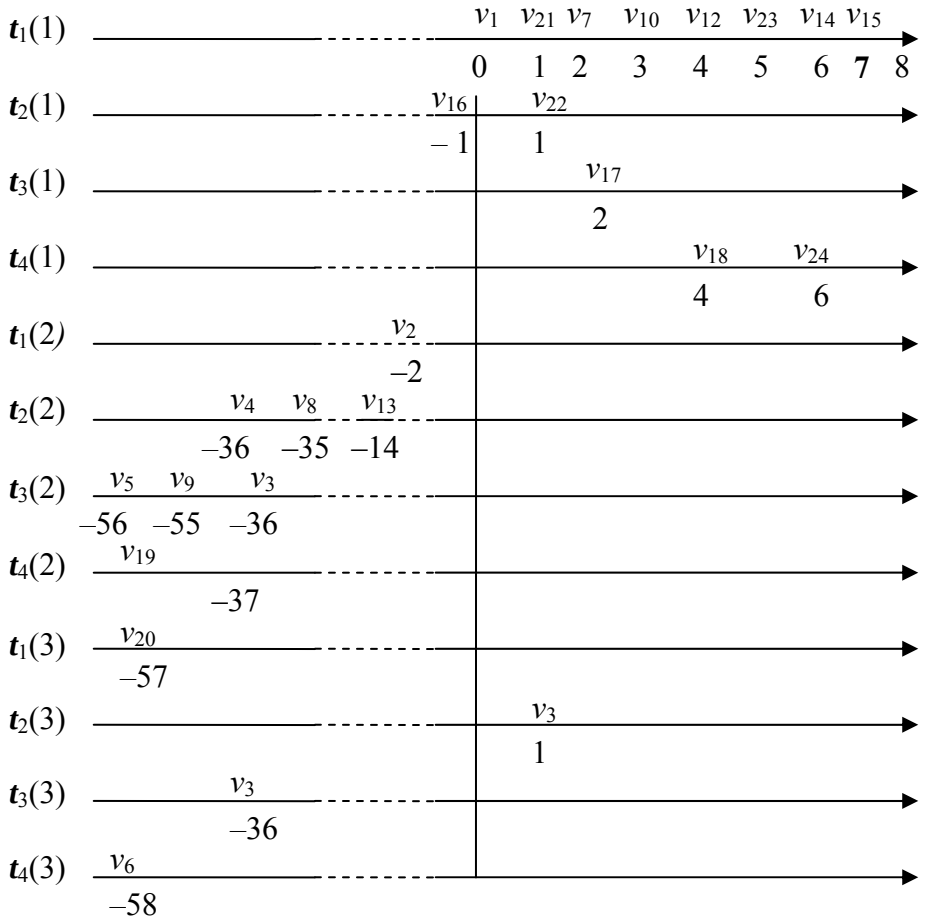

Рис. 14.4. Формирование вектора временной развертки усеченных путей ВГА: v_1–v_{24} – вершины графа, $t_i(\gamma)$ – вектор временной развертки i-го пути уровня γ

Тогда ВВР вычислительного графа алгоритма будет следующим:

$\boldsymbol{t}$ = (0, –2, –38, –34, –56, –58, 2, –33, –55, 3, –38, 5, –14, 7, 8, –1, 2, 4, –37, –57, 1, 1, 6, 6).

Нормированный ВВР:

$\boldsymbol{t}_0$ = (58, 56, 20, 24, 2, 0, 60, 25, 3, 61, 20, 63, 44, 65, 67, 57, 60, 62, 21, 1, 59, 59, 64, 64).

14.3. Задание

1. Разработать программу формирования ВВР. Исходные данные:

– усеченные пути ВГА $\{\boldsymbol{L}_j(\gamma)\}$;

– способы отображения усеченных путей на архитектуру вычислительной системы (конвейерные или бесконвейерные);

– вектор шагов дискретизации $\Delta\boldsymbol{t}$;

– ступени конвейера для конвейеризированных путей $\{V(L_k(\gamma), \mu)\}$;

– вектор реализации τ.

2. Проверить работу программы на тестовом примере.

Лабораторная работа № 15

ВТОРАЯ ПРОВЕРКА РЕАЛИЗУЕМОСТИ ВЫЧИСЛИТЕЛЬНОЙ СИСТЕМЫ РЕАЛЬНОГО ВРЕМЕНИ

Целью лабораторной работы является изучение дополнительных условий реализуемости вычислительной системы реального времени.

15.1. Условие реализуемости вычислительной системы при свертке вершин ВГА

Свертка вершин ВГА при формировании графа вычислительной структуры может привести к невозможности функционирования системы в реальном времени в силу невыполнения одного из необходимых условий возможности свертки вершин графа – выполнения функциональными устройствами, назначенными вершинам, входящим в соответствующее множество свертываемых вершин, своих функций на непересекающихся интервалах времени. Установление временных соотношений для ФУ системы с помощью найденного вектора временной развертки позволяет выявить подобные ситуации до выполнения заключительных процедур синтеза ВС. В основу проверки реализуемости ВС на данном этапе проектирования может быть использовано следующее положение.

УТВЕРЖДЕНИЕ 15.1. Необходимым условием реализуемости ВСРВ является выполнение для всех вершин ВГА, входящих в каждое из множеств свертываемых вершин для рассматриваемого проектного варианта, условия:

$$\forall (i, j) \in V(=S): t_j > t_i: t_j - (t_i + \tau_i) \geq 0. \qquad (15.1)$$

Т. е. операция, отождествленная с i-й вершиной, входящей во множество S, должна быть закончена не позднее начала выполнения операции i-й вершиной из того же множества S.

В случае невыполнения условия (15.1) хотя бы для одной пары вершин, входящих в одно из подмножеств $\{S\}$ из текущего кортежа C, текущий вариант исключается из дальнейшего рассмотрения, и необходимо перейти к следующему из отобранных вариантов.

15.2. Пример

Задача. Проверить возможность реализуемости проектного решения, характеризующегося следующими атрибутами:

а) множества свертываемых вершин

$$S_1 = \{7, 13\}, S_2=\{5, 8, 11\};$$

б) вектор временной развертки

$$\boldsymbol{t} = (3, 1, 0, 7, 10, 2, 4, 16, 8, 12, 18, 15, 14, 23);$$

в) вектор реализации

$$\boldsymbol{\tau} = (2, 6, 1, 4, 6, 3, 7, 6, 4, 12, 6, 8, 7, 2).$$

Решение. Для вершин, входящих во множество S_1, из (15.1) получим:

$$t_{13} – (t_7 + \tau_7) = 14 – (4 + 7) = 3 > 0.$$

Таким образом, свертка вершин множества S_1 не приводит к нереализуемости ВСРВ.

Рассмотрим выполнение условия реализуемости при свертке вершин множества S_2.

$$t_8 – (t_5 + \tau_5) = 16 – (10 + 6) = 0;$$

$$t_{11} – (t_5 + \tau_5) = 18 – (10 + 6) = 2 > 0;$$

$$t_{11} – (t_8 + \tau_8) = 18 – (16 + 6) = –4.$$

Из полученного результата следует невозможность свертки вершин v_8 и v_{11}, таким образом, данный априорный вариант ВСРВ является нереализуемым (неработоспособным).

15.3. Задание

1. Разработать программу проверки реализуемости ВСРВ. Исходные данные:

– множества подмножеств свертываемых вершин $\{S\}$;

– вектор временной развертки $\boldsymbol{t}$;

– вектор реализации $\boldsymbol{\tau}$.

Результат работы программы – сообщение о реализуемости/нереализуемости текущего проектного решения.

2. Проверить работу программы на тестовом примере.

Лабораторная работа № 16

ПОСТРОЕНИЕ ВЫЧИСЛИТЕЛЬНОГО ГРАФА АЛГОРИТМА С РЕГИСТРОВЫМИ ФАЙЛАМИ

Целью лабораторной работы является изучение принципов построения графа вычислительной структуры с регистровыми файлами (ГВСРФ).

16.1. Формирование вектора требований к памяти

ОПРЕДЕЛЕНИЕ 16.1. *Вектором требований к памяти* называется вектор-строка $\boldsymbol{P}$ размером J, где J – количество дуг ВГА вида

$$\boldsymbol{P} = -(A'\boldsymbol{t} + \boldsymbol{\tau}), \tag{16.1}$$

где A' – транспонированная матрица инциденций; $\boldsymbol{t}$ и $\boldsymbol{\tau}$ – вектор временной развертки и вектор реализации соответственно.

УТВЕРЖДЕНИЕ 16.1. Элемент p_i вектора временной развертки определяется как разность

$$p_j = t_l - t_k - \tau_j\,, \tag{16.2}$$

где t_l и t_k – элементы ВВР для вершин k и l, связанных дугой $j = (k, l)$; τ_j – элемент вектора реализации, соответствующий j-й дуге.

16.2. Корректность проектирования вычислительной системы

УТВЕРЖДЕНИЕ 16.2. Необходимым и достаточным условием реализуемости вычислительного графа алгоритма вычислительной системой в реальном времени является отсутствие в векторе требований к памяти отрицательных элементов.

Действительно, наличие отрицательных элементов вектора $\boldsymbol{P}$ эквивалентно наличию неравенства вида $t_l - t_k < \tau_k$, что противоречит условию реализации пути $L = (v_k, v_l)$ в реальном времени. Так как методы, положенные в основу всех рассмотренных ранее про-

цедур синтеза, базировались на условии реализуемости ВС в реальном времени, наличие отрицательных элементов вектора $\boldsymbol{P}$ свидетельствует о некорректности разработанных алгоритмов или о машинном сбое.

16.3. Построение ВГА с регистровыми файлами

ОПРЕДЕЛЕНИЕ 16.2. *Вычислительным графом алгоритма с регистровыми файлами* называется граф, формируемый из вычислительного графа алгоритма путем выполнения операции добавления вершин для дуг с номерами, равными индексам положительных элементов вектора $\boldsymbol{P}$.

Добавленным вершинам назначается блок памяти с верхней границей доступа $\tau_i \le p_k$, где p_k – k-й элемент вектора $\boldsymbol{P}$. В результате имеем преобразование:

$$G^{\text{ВГА}} \xrightarrow{\text{ДВ}} G^{\text{РФ}}=\left(V^{\text{РФ}}, E^{\text{РФ}}\right),$$

при этом справедливо утверждение 2.13.

Полученный граф характеризуется спецификацией, вектором назначения, вектором реализации, вектором временной развертки.

16.4. Пример

Вектор требований к памяти для ВГА, изображенном на рис. 12.5 при нормализованном векторе временной развертки 12.1 и векторе реализации 12.2 будет иметь следующий вид:

$\boldsymbol{P}$ = (0, 0, 39, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 4, 7, 0, 9, 0, 4, 0, 0, 0, 0, 0).

Вычислительный граф алгоритма с регистровыми файлами, полученный из ВГА, изображенного на рис. 12.5, при векторе требований к памяти (16.3), представлен на рис. 16.1. Спецификация его вершин v_1–v_{24} соответствует спецификации вершин ВГА, вершины v_{25}–v_{29} отождествляется с операцией хранения.

Из векторов назначения выберем следующий:

$$\boldsymbol{R}^{\text{РФ}}= (19, 19, 19, 19, 19, 19, 21, 5, 10, 8, 7, 11, 21, 20,\\ 13, 13, 13, 13, 13, 16, 16, 16, 16, 13, 13, 13, 13, 13), \quad (16.3)$$

соответствующий данному вектору назначения вектор реализации:

$$\boldsymbol{\tau}^{\text{РФ}} = (1, 1, 1, 1, 1, 1, 1, 1, 15, 12, 2, 1, 1, 1, 16, 1, 1,\\ 2, 2, 2, 2, 2, 1, 1, 1, 1, 2, 2, 2, 2, 2). \quad (16.4)$$

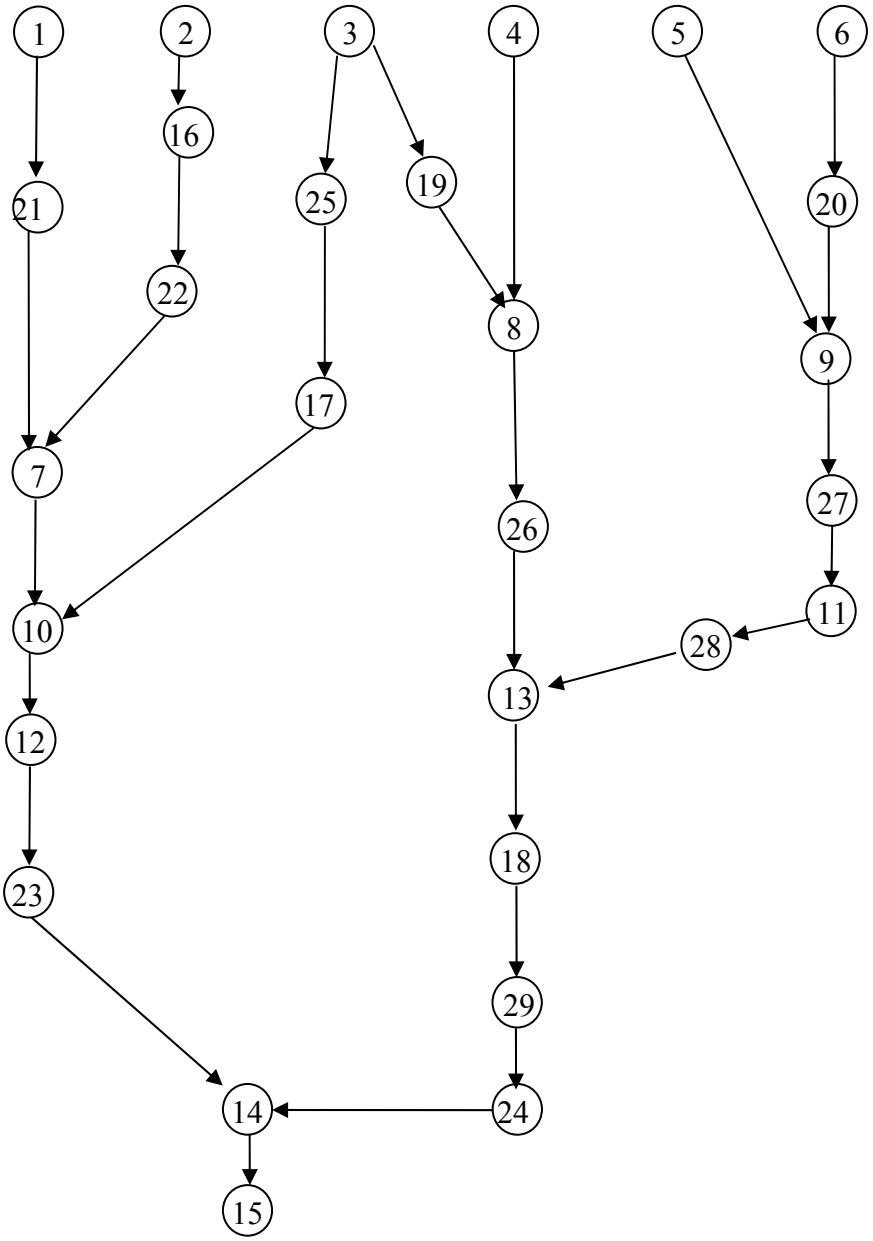

Рис. 16.1. Вычислительный граф алгоритма с регистровыми файлами

Координаты вектора временной развертки для вершин v_{25}–v_{29} определяются в соответствии с утверждением (13.1):

$$\boldsymbol{t}^{\text{РФ}} = (58, 56, 20, 22, 20, 60, 23, 3, 61, 20, 63, 42, 65, 67, 57, 60, 62, 21,1, 59, 64, 21, 38, 15, 31, 58). \quad (16.5)$$

В результате выполнения процедуры формируются: новое множество свертываемых вершин $S_4 = \{25, 17\}$, представляющее собой две последовательно соединенные вершины, реализуемые одним блоком памяти, и по той же причине множество $S_4 = \{29, 18\}$.

16.5. Задание

1. Разработать программу формирования вектора требований к памяти. Исходные данные:

– вычислительный граф алгоритма $G^{\text{ВГА}}$;

– вектор назначения $\boldsymbol{R}^{\text{ВГА}}$;

– вектор реализации $\boldsymbol{\tau}^{\text{ВГА}}$.

2. Разработать программу формирования вычислительного графа алгоритма с регистровыми файлами. Исходные данные:

– вычислительный граф алгоритма $G^{\text{ВГА}}$ ($V^{\text{ВГА}}$, $E^{\text{ВГА}}$);

– вектор требований к памяти $\boldsymbol{P}$.

Результат выполнения программы:

– вычислительный граф алгоритма с регистровыми файлами $G^{\text{РФ}}$ ($V^{\text{РФ}}$, $E^{\text{РФ}}$);

– вектора назначения $\boldsymbol{R}^{\text{РФ}}$;

– вектор реализации $\boldsymbol{\tau}^{\text{РФ.}}$

3. Проверить работу программы на тестовом примере, заданном преподавателем.

Лабораторная работа № 17
ФОРМИРОВАНИЕ ГРАФА ВЫЧИСЛИТЕЛЬНОЙ СТРУКТУРЫ

Целью лабораторной работы является изучение принципов построения графа вычислительной структуры.

17.1. Определение графа вычислительной структуры

Граф вычислительной структуры (ГВС) является последним графом, сформированным путем последовательных преобразований исходного графа (графа вычислительного алгоритма).

ОПРЕДЕЛЕНИЕ 17.1. *Графом вычислительной структуры* называется граф, отображающий структуру физической реализации вычислительной системы.

Вершинам ГВС соответствуют физические компоненты, дугам – связи между этими компонентами. ГВА $G^{\text{ВС}} = (V^{\text{ВС}}, E^{\text{ВС}})$ формируется из вычислительного графа алгоритма с регистровыми файлами путем последовательного выполнения операции простого элементарного гомоморфизма над каждым из множеств свертываемых вершин, при этом подмножество свертываемых вершин $S_k = (v_i, v_j, \ldots, v_l)$ во множестве $V^{\text{РФ}}$ заменяется во множестве $V^{\text{ВС}}$ одной вершиной. Множество дуг $E^{\text{ВС}}$ строится следующим способом. Если во множестве E дуга не инцидентна хотя бы одной из вершин $S_h = (v_i, v_j, \ldots, v_l)$, то она переносится во множество E. Если же дуга инцидентна хотя бы одной из этих вершин, то она формируется в E заменой вершин $(v_i, v_j, \ldots, v_l)$ новой вершиной.

Полученный граф характеризуется спецификацией, вектором назначения, вектором реализации, функционалом временной развертки.

ГВС учитывает возможности многократного срабатывания некоторых ФУ на одном цикле функционирования ВС. Данный граф характеризуется вектором назначения $\boldsymbol{R}^{BC}$, спецификацией вершин $\{\varphi_i(v_i)\}$, таблицей переходов, устанавливающей соответствие между вершинами вычислительного графа алгоритма с регистровыми файлами (прообразами) и графа вычислительной структуры (образами):

$$\{v_k \leftrightarrow \{v_e\}\},$$

где v_k – k-я вершина ГВА РФ; $\{v_e\}$ – множество образов вершины v_k во множестве вершин ГВС.

17.2. Пример

Задание. Дан граф вычислительного алгоритма с РФ из примера 16.4 (см. рис. 16.1), множество свертываемых вершин

$$S_1 = \{7, 14\},\ S_2 = \{21, 23\},\ S_3 = \{22, 24\},$$
$$S_4 = \{25, 17\},\ S_5 = \{29, 18\}, \qquad (17.1)$$

вектор назначения (16.1), вектор временной развертки (16.2). Построить граф вычислительной структуры.

Решение. Граф вычислительной структуры имеет вид, представленный на рис. 17.1, спецификация его вершин – в табл. 17.1. Соответствие вершин графа вычислительной структуры вершинам графа алгоритма с регистровыми файлами иллюстрируется табл. 17.2.

Таблица 17.1

Спецификация вершин графа вычислительной структуры

V_j	V_1	V_2	V_3	V_4	V_5	V_6	V_7	V_8
Φ_j	Ввод x_1	Ввод x_2	Ввод x_3	Ввод x_4	Ввод x_5	Ввод x_6	+	*

Продолжение табл. 17.1

V_j	V_9	V_{10}	V_{11}	V_{12}	V_{13}	V_{14}	V_{15}	V_{16}
Φ_j	+	f_2	f_1	f_1	f_2	*Xp*	Вывод y	*Mx*

Окончание табл. 17.1

V_j	V_{17}	V_{18}	V_{19}	V_{20}	V_{21}	V_{22}	V_{23}	V_{24}
Φ_j	*Mx*	*Xp*	*Xp*	*Xp*	*Xp*	*Xp*	*Xp*	*Xp*

Таблица 17.2

Взаимное соответствие вершин ГВС и ВГАРФ

$V^{ВС}$	V_1	V_2	V_3	V_4	V_5	V_6	V_7	V_8	V_9	V_{10}	V_{11}	V_{12}	V_{13}
$V^{РФ}$	V_1	V_2	V_3	V_4	V_5	V_6	V_7, V_{14}	V_8	V_9	V_{10}	V_{11}	V_{12}	V_{13}

Окончание табл. 17.2

$V^{ВС}$	V_{14}	V_{15}	V_{16}	V_{17}	V_{18}	V_{19}	V_{20}	V_{21}	V_{22}	V_{23}	V_{24}
$V^{РФ}$	V_{16}	V_{15}	V_{21}, V_{23}	V_{22}, V_{24}	V_{17}, V_{25}	V_{19}	V_{20}	V_{26}	V_{27}	V_{28}	V_{18}

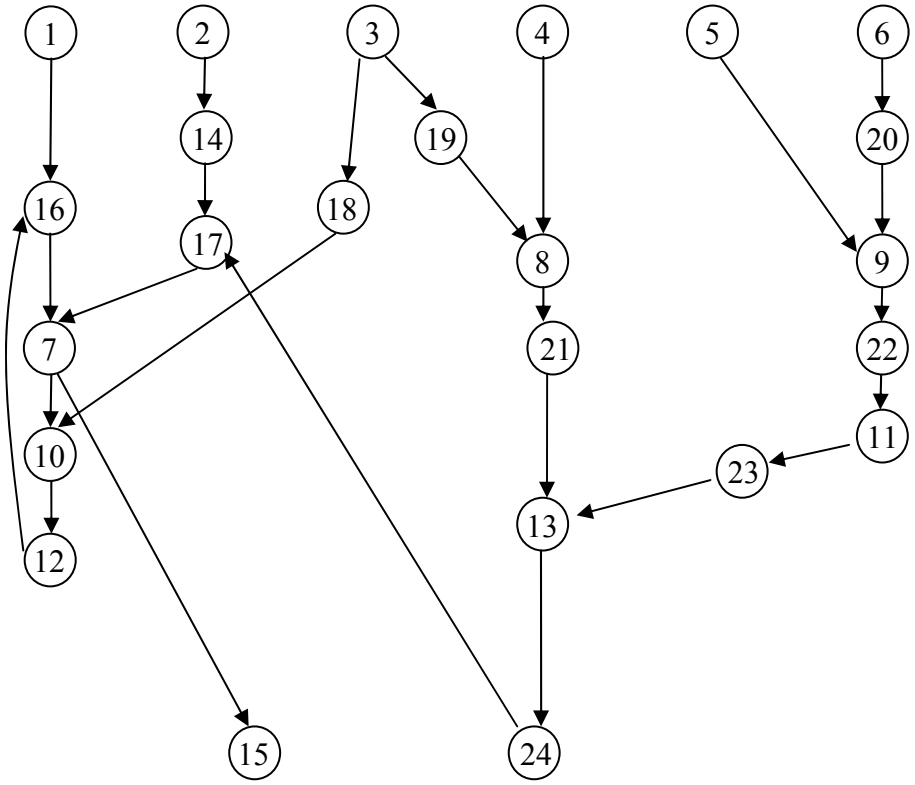

Рис. 17.1. Граф вычислительной структуры

Вектор назначения для полученного графа $\boldsymbol{R}^{\text{ВС}}$ = (19, 19, 19, 19, 19, 19, 21, 5, 2, 10, 8, 7, 11, 13, 20, 16, 13, 13, 13, 13, 13, 13, 13).

17.3. Задание

1. Разработать программу формирования графа вычислительной структуры. Исходные данные:

– граф вычислительного алгоритма $G^{\text{РФ}}$ ($V^{\text{РФ}}$, $E^{\text{РФ}}$);

– множество свертываемых вершин $\{S_m\}$;

– вектор назначения $\boldsymbol{R}^{\text{РФ}}$.

2. Протестировать программу на примере, заданном преподавателем.

Лабораторная работа № 18

ФОРМИРОВАНИЕ УПРАВЛЯЮЩИХ ПАРАМЕТРОВ ВЫЧИСЛИТЕЛЬНОЙ СИСТЕМЫ

Целью лабораторной работы является изучение принципов формирования параметров вычислительных систем.

18.1. Определения управляющих параметров

ОПРЕДЕЛЕНИЕ 18.1. *Функционалом временной развертки* графа вычислительной структуры, содержащего J вершин, называется совокупность из J равенств вида:

$$\Phi\big(t(j,q)\big) = t(j,q) + \Delta t\left(\gamma_j\right)\cdot k_j,$$
$$j = \overline{1, J},\ k_j = 0, 1, 2, 3, \ldots \qquad (18.1)$$
$$q = \overline{1, Q_j},$$

где $t(j, q)$ – требуемые моменты включения i-го ФУ на первом цикле функционирования, равные координатам вектора временной развертки вычислительного графа алгоритма с регистровыми файлами, соответствующие свертываемым вершинам этого графа при формировании вершины v_i графа вычислительной структуры; $\Delta t(\gamma_i)$ – шаг дискретизации для уровня временной иерархии вершины v_i; k_i – коэффициент, задающий периодичность процесса; Q_i – количество срабатываний j-го ФУ на одном цикле функционирования.

Функционал временной развертки отображает возможности многократного срабатывания ФУ на одном цикле их функционирования (первое слагаемое в каждом из уравнений) и периодичность работы ФУ (второе слагаемое).

Вторым синхронизационным параметром является такт вычислительной структуры.

ОПРЕДЕЛЕНИЕ 18.2. *Тактом вычислительной структуры* τ_c называется наибольший общий делитель (НОД) для всех интервалов времени между любыми событиями в вычислительной структуре:

$$\tau_c = \text{НОД}\left(t_j - t_k\right) \text{ при } \forall k, j \in i,\ i = 1,\ \ldots,\ l, \qquad (18.2)$$

где t_j, t_k – координаты ВВР; I – количество элементов ВВР.

18.2. Пример

Определить функционал временной развертки для графа вычислительной структуры из примера 17.2, при следующих его атрибутах:

– вектор временной развертки ***t*** (16.2);

– множества свертываемых вершин (17.1);

– таблица переходов (взаимного соответствия вершин), заданная в табл. 17.2;

– шаги дискретизации (5.1).

На основании вектора временной развертки ВГАРФ, табл. 17.2 и исходных данных сформируем функционал временной развертки для вычислительной системы:

$$\Phi\left(t\left(1, q_1\right)\right) = 58 + 10k_1;$$

$$\Phi\left(t\left(2, q_2\right)\right) = 56 + 20k_2;$$

$$\Phi\left(t\left(3, q_3\right)\right) = 20 + 40k_3;$$

$$\Phi\left(t\left(4, q_4\right)\right) = 22 + 40k_4;$$

$$\Phi\left(t\left(5, q_5\right)\right) = 2 + 20k_5;$$

$$\Phi\left(t\left(6, q_6\right)\right) = 0 + 40k_6;$$

$$\Phi\left(t\left(7, q_7\right)\right) = (60, 65) + 10k_7;$$

$$\Phi\left(t\left(8, q_8\right)\right) = 23 + 20k_8;$$

$$\Phi\left(t\left(9, q_9\right)\right) = 3 + 20k_9;$$

$$\Phi\left(t\left(10, q_{10}\right)\right) = 61 + 10k_{10};$$

$$\Phi\left(t\left(11, q_{11}\right)\right) = 20 + 20k_{11};$$

$$\Phi\left(t\left(12, q_{12}\right)\right) = 63 + 10k_{12};$$

$$\Phi\left(t\left(13, q_{13}\right)\right) = 40 + 20k_{13};$$

$$\Phi\left(t\left(14, q_{14}\right)\right) = 57 + 20k_{14};$$

$$\Phi\left(t\left(15, q_{15}\right)\right) = 67 + 10k_{15};$$

$$\Phi\left(t\left(16, q_{16}\right)\right) = (59, 64) + 10k_{16};$$

$$\Phi\left(t\left(17, q_{17}\right)\right) = (59, 64) + 10k_{17};$$

$$\Phi\left(t\left(18, q_{18}\right)\right) = 21 + 20k_{18};$$

$$\Phi\left(t\left(19, q_{19}\right)\right) = 21 + 40k_{19};$$

$$\Phi\left(t\left(20, q_{20}\right)\right) = 1 + 40k_{20};$$

$$\Phi\left(t\left(21, q_{21}\right)\right) = 38 + 20k_{21};$$

$$\Phi\left(t\left(22, q_{22}\right)\right) = 15 + 20k_{22};$$

$$\Phi\left(t\left(23, q_{23}\right)\right) = 31 + 20k_{23};$$

$$\Phi\left(t\left(24, q_{24}\right)\right) = 58 + 20k_{24}.$$

Такт вычислительной структуры

$$\tau_c = \min_{k,\, j \in \{i\}} \{t_k - t_j\} = 1. \qquad (18.3)$$

18.3. Задание

1. Разработать программу формирования параметров управления (функционала временной развертки и такта вычислительной структуры). Исходные данные:

– граф вычислительной структуры $G^{\text{ВС}} = (V^{\text{ВС}}, E^{\text{ВС}})$;

– вектор временной развертки $\boldsymbol{t}^{\text{РФ}}$;

– таблица переходов $\{v_k \to \{v_i\}\}$;

– вектор шагов дискретизации Δ;

– множества вершин УВИ $\{v(\gamma)\}$.

2. Проверить работу программы на примере, заданном преподавателем.

Лабораторная работа № 19

ВЫБОР ИЗ МНОЖЕСТВА АЛЬТЕРНАТИВНЫХ ВАРИАНТОВ, ОПТИМАЛЬНЫХ ПО ЗАДАННЫМ КРИТЕРИЯМ

Целью лабораторной работы является изучение логико-комбинаторного подхода к решению оптимизационных задач при синтезе вычислительных систем.

19.1. Логико-комбинаторный метод оценки эффективности проектных решений

В соответствии с одним из центральных положений рассмотренной методологии синтеза вычислительных систем реального времени, для каждого вектора из множества $\{R\}$ должна выполняться процедура определения множеств свертываемых вершин. Предположим, что граф вычислительного алгоритма или нагруженный граф (так называемый граф базовой структуры) содержит дуги (i, k), (j, k), (l, n), (m, n), (s, t), (q, t) (рис. 19.1), при этом в соответствии с выбранным на одном из более ранних этапов синтеза вектором назначения выполняется равенство его элементов $r_k = r_n = r_t$, т. е. $v_k \leftrightarrow z_j$, $v_n \leftrightarrow z_j$, $v_t \leftrightarrow z_j$.

Тогда возможны следующие варианты множества свертываемых вершин:

$S_0^{(1)} = 0$ – пустое множество (отсутствие свертки);

$S_1^{(1)} = (k, n)$; $S_2^{(1)} = (k, t)$;

$S_3^{(1)} = (t, n)$; $S_4^{(1)} = (k, n, t)$.

В общем случае справедливо следующее утверждение: если существуют (i, k), (i, n), …, (s, t) и $k, n, \ldots, t \subset S$, то возможно преобразование:

$$(i, k) \overset{\text{ДВ}}{\rightarrow} (i, v), (v, k); (l, n) \overset{\text{ДВ}}{\rightarrow} (l, w), (w, n); \ldots;$$

$$(s, t) \overset{\text{ДВ}}{\rightarrow} (s, p); (p, t), (v, w, p) = S^1,$$

т. е. добавленные вершины образуют новые множества свертываемых вершин, каждой из которых назначается ФЭ одного типа (один экземпляр $a_e^{(k)}$). ДВ – операция добавления вершины.

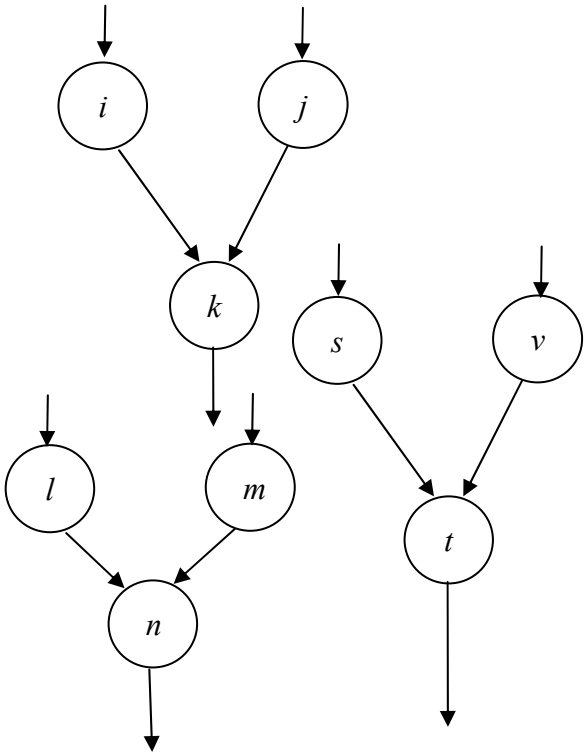

Рис. 19.1. Фрагмент ГВА

Фрагмент преобразованного ГВА для приведенного выше примера в этом случае будет иметь вид, представленный на рис. 19.2.

В силу вышесказанного следует, что каждый из векторов $\boldsymbol{R}$ является порождающим для нового множества векторов назначения и соответствующих им структур, что существенно увеличивает количество альтернативных вариантов проектируемой системы. Таким образом, одной из важнейших задач, решаемых при синтезе вычислительной системы, является задача выбора из множества работоспособных синтезированных вариантов оптимального по заданным критериям. В дальнейшем будет показано, что данная задача может решаться на одном из ранних этапов синтеза.

Выбор оптимального варианта в простейшем случае может решаться при следующей постановке оптимизационной задачи:

$$\boldsymbol{b} \rightarrow \min,\ \boldsymbol{A}\mathrm{z}_{\boldsymbol{p}} \le \alpha, \qquad (19.1)$$

где $\boldsymbol{b}$ – вектор булевых переменных, обозначающих отдельные элементы из множества Z; $\boldsymbol{P} = (p_1, \ldots, p_n)$ – вектор весов элементов

$Z_i \in Z$; $\boldsymbol{A}\boldsymbol{z_p} \le \alpha$ – система ограничений для атрибутов проектируемого объекта.

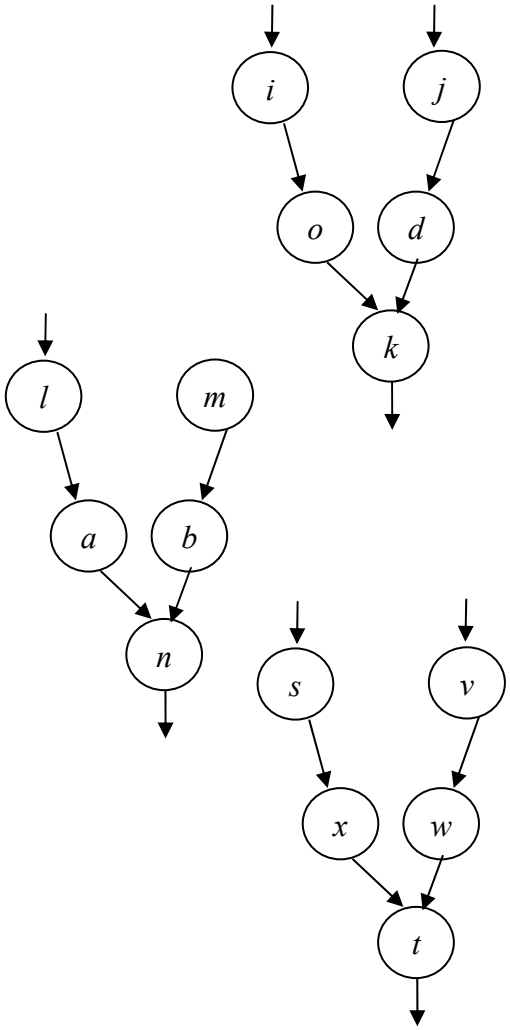

Рис. 19.2. Фрагмент преобразованного ГВА

Задача формализации этого важнейшего этапа проектирования сложных технических объектов (выбор оптимального варианта структуры) может быть решена с помощью аппарата *характеристических булевых* (*логических*) *функций*.

ОПРЕДЕЛЕНИЕ 19.1. Характеристической булевой функцией (ХБФ) множества альтернативных вариантов называется булева функция $f(z_1, \ldots, z_n)$, простая импликанта которой равна единице тогда и только тогда, когда ее аргументы $z_1^1, \ldots, z_1^n$ представляют собой правильный вариант структуры.

Установим взаимнооднозначное соответствие между ФЭ и аргументами характеристической функции:

$$z_i \leftrightarrow b_i.$$

Тогда перечисление всех вариантов структуры проектируемого объекта можно представить в виде дизъюнкции всех простых импликант ХБФ, т. е. сокращенной дизъюнктивной нормальной формы (сокр. д. н. ф.):

$$F(b_1^{q1},\ldots,b_N^{q}) = f_1 \vee f_2 \vee \ldots \vee f_N = (b_1^{(1)} \wedge b_2^{(1)} \wedge \ldots \wedge b_{N1}^{(1)}) \vee (b_1^{(2)} \wedge b_2^{(2)} \wedge \ldots \wedge b_i^{(2)} \wedge b_j^{(2)} \wedge \ldots \wedge b_{N2}^{(2)}) \vee \vee (b_1^{(Q)} \wedge \ldots \wedge b_{NQ}^{(Q)}).$$

Операцией конъюнкции связываются переменные, включаемые совместно в один из вариантов структуры, функция дизъюнкции соответствует операции выбора одного из альтернативных вариантов.

С ростом числа альтернативных вариантов целесообразно использовать особенные скобочные нормальные формы (о. ск. н. ф.) характеристической булевой функции. Для ее формирования достаточно выполнить операцию факторизации и использовать правила булевой алгебры, в частности: $x \wedge x \wedge \ldots \wedge x = x$. В качестве примера рассмотрим минимизацию произвольной логической функции:

$$F(b_1^{(q)},\ldots,b_{Nq}^{(q)}) = (b_\alpha^{(1)} \wedge \ldots \wedge b_\beta^{(1)} \wedge b_i^{(1)} \wedge b_\gamma \wedge \ldots \wedge b_i^{(2)} \wedge \ldots \wedge b_i^{(3)}) \vee (b_\alpha^{(1)} \wedge \ldots \wedge b_\beta^{(1)} \wedge b_i^{(1)} \wedge b_j^{(1)} \wedge \mathrm{b}_\gamma \wedge \ldots \wedge b_i^{(1)} \wedge b_j^{(1)} \wedge \ldots \wedge b_i^{(2)}) \vee (b_\alpha^{(1)} \wedge \ldots \wedge b_\beta^{(1)} \wedge b_i^{(1)} \wedge b_j^{(1)} \wedge \ldots \wedge b_i^{(2)} \wedge b_\gamma \wedge \ldots \wedge b_i^{(1)} \wedge b_j^{(1)} \wedge \ldots) \vee (b_\alpha \wedge \ldots \wedge b_\beta \wedge b_i^{(1)} \wedge b_j^{(1)} \wedge b_\gamma \ldots \wedge \ldots \wedge b_i^{(1)} \wedge \ldots \wedge b_j^{(1)} \wedge \ldots b_i^{(1)} \wedge \ldots \wedge b_j^{(1)}) = b_\alpha \wedge b_\beta \wedge \ldots \wedge b_\gamma \wedge [(b_i^{(1)} \wedge b_i^{(2)} \wedge b_i^{(3)}) \vee (b_i^{(1)} \wedge b_j^{(1)} \wedge b_i^{(1)} \wedge b_j^{(1)} \wedge b_i^{(2)}) \vee (b_i^{(1)} \wedge b_j^{(1)} \wedge b_i^{(2)} \wedge b_i^{(1)} \wedge b_j^{(1)}) \vee (b_i^{(1)} \wedge b_j^{(1)} \wedge b_i^{(1)} \wedge b_j^{(1)} \wedge b_i^{(1)} \wedge b_j^{(1)})] = b_\alpha : b_\beta \wedge \ldots \wedge b_\gamma \wedge [(b_i^{(1)} \wedge b_i^{(2)} \wedge b_i^{(3)}) \vee \wedge (b_i^{(1)} \wedge b_j^{(1)} \wedge b_i^{(2)}) \vee (b_i^{(1)} \wedge b_j^{(1)})].$$

Данное выражение содержит не исчерпывающее количество всех возможных вариантов из NP-полного перебора; некоторые импликанты могут дублировать друг друга, представляя, тем не менее, разные варианты структуры.

Один из методов решения проблемы выбора оптимального варианта структуры включает в себя следующие основные этапы.

1. Установление взаимнооднозначного соответствия между ФУ проектируемой системы и булевыми переменными $z_i \leftrightarrow b_i$.

2. Представление множества альтернативных вариантов структуры проектирования системы в виде дизъюнкции всех простых импликант характеристической булевой функции (ХБФ), т. е. сокращенной дизъюнктивной нормальной формы (сокр. д. н. ф)

3. Преобразование сокр. д. н. ф. →о. ск. н.ф.

4. Переход к арифметической форме представления альтернативных вариантов путем замены аргументов и операций в формуле ХБФ:

– $b_i \to p_i$; p_i – вес ФУ, соответствующего i-й булевой переменной.

– $b_i \wedge b_j \to p_i + p_j$;

– $b_i \vee b_j \to \min(p_i, p_j)$.

5. Определение min (P_f) и идентификации варианта оптимальной структуры.

19.2. Пример

Процедура построения варианта структуры с минимальным суммарным весом ФУ – элементов, составляющих структуру, в соответствии с п. 4 рассмотренного в подразделе 19.1 алгоритма может быть осуществлена следующим образом:

– замена в формуле ХБФ – $b_i \to p_i$;

– замена логических функций на арифметические: конъюнкция → арифметическое сложение, дизъюнкция → функция взятия минимума.

Предположим, о. ск. н. ф. имеет вид:

$$f(z_1, \ldots, z_{13}) = z_1 z_3[z_{13}(z_7 \vee z_8 \vee z_{12} z_6) \vee z_6 z_{11}(z_7 \vee z_8 \vee z_{10} \vee z_{12}) \vee$$

$$\vee z_{10} \vee z_{12}) \vee (z_4 \vee z_5 z_2 \vee z_9)(z_7 \vee z_8 \vee z_{12} z_6 \vee z_{10})].$$

Пусть значения веса функциональных элементов, выраженные в условных единицах, равны $p_1 = 0$, $p_2 = 5$, $p_3 = 4$, $p_4 = 7$, $p_5 = 8$, $p_6 = 3$, $p_7 = 6$, $p_8 = 4$, $p_9 = 6$, $p_{10} = 4$, $p_{11} = 5$, $p_{12} = 3$. Тогда для фиксации вариантов подструктур, получающихся в процессе вычислений, значениям весов приписываются сложные индексы в виде подформул соответствующих подструктур. Ради краткости в этих подформулах указываются только индексы переменных – z_i ($I = 1, \ldots, n$) вместо самих переменных. Тогда можно записать:

$$\min p_f = p_3 + \min(p_{13} + \min(p_7, p_8, p_{12} + p_6), p_6 + p_{11} + \min(p_7, p_8, p_{10}, p_{12}),$$

$$\min(p_4, p_5 + p_2, p_9) + \min(p_7, p_8, p_{12} + p_6, p_{10})) =$$

$$= 4_3 + \min(3_{13} + 4_8, 3_6 + 5_{11} + 4_{8\vee 10}, 6_9 + 4_{8\vee 10}) =$$

$$= 4_3 + \min(7_{13\wedge 8}, 12_{6\wedge 11(8\vee 10)}, 10_{9(8\vee 10)}) = 4_3 + 7_{13\wedge 8} = 11_{3\wedge 13\wedge 8}.$$

Таким образом, минимальный вес $p = 11$ имеет вариант $z_1 z_3 z_{13} z_8$.

19.3. Задание

1. Разработать программу выбора оптимального варианта проектируемой вычислительной системы. Исходные данные:

– граф вычислительной системы $\{G^{\text{ВС}}\}$;

– множество векторов назначения $\{\boldsymbol{R}^{\text{ВС}}\}$;

– весовые коэффициенты $p = (p_1, p_2, \ldots, p_n)$.

2. Проверить работу программы на тестовом примере, заданном преподавателем.

ЛИТЕРАТУРА

1. Алгоритмы: построение и анализ / Т. Кормен [и др.]. – М.: СПб.: Киев: Издательский дом «Вильнюс», 2005. – 1291 с.

2. Воеводин, В. В. Математические модели и методы в параллельных процессах / В. В. Воеводин – М.: Наука. Главное изд-во физ.-мат. литературы, 1989. – 296 с.

3. Касьянов, В. Н. Графы в программировании: обработка, визуализация и применение / В. Н. Касьянов, В. А. Евстигнеев – СПб.: БХВ-Петербург, 2003. – 1104 с.

4. Кобайло, А. С. Теория синтеза вычислительных систем реального времени: монография / А. С. Кобайло. – Минск: БГТУ, 2010. – 256 с.

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ 3

Лабораторная работа № 1
ФОРМИРОВАНИЕ ГРАФА ВЫЧИСЛИТЕЛЬНОГО АЛГОРИТМА 5

Лабораторная работа № 2
СИНТЕЗ МАТЕМАТИЧЕСКИХ МОДЕЛЕЙ С ЗАДАННЫМИ СВОЙСТВАМИ 13

Лабораторная работа № 3
СИНТЕЗ МАТЕМАТИЧЕСКИХ МОДЕЛЕЙ С ТРЕБУЕМЫМИ СВОЙСТВАМИ 17

Лабораторная работа № 4
ОПРЕДЕЛЕНИЕ ВСЕХ ПОЛНЫХ ПУТЕЙ ГРАФА ВЫЧИСЛИТЕЛЬНОГО АГОРИТМА 25

Лабораторная работа № 5
НАЗНАЧЕНИЕ УРОВНЕЙ ВРЕМЕННОЙ ИЕРАРХИИ ВЕРШИНАМ ГРАФА ВЫЧИСЛИТЕЛЬНОГО АЛГОРИТМА 27

Лабораторная работа № 6
НАЗНАЧЕНИЕ ФУНКЦИОНАЛЬНЫХ УСТРОЙСТВ ВЕРШИНАМ ГРАФА БАЗОВОЙ СТРУКТУРЫ 31

Лабораторная работа № 7
ФОРМИРОВАНИЕ ГРАФА БАЗОВОЙ СТРУКТУРЫ С БУФЕРНОЙ ПАМЯТЬЮ 37

Лабораторная работа № 8
ФОРМИРОВАНИЕ УСЕЧЕННЫХ ПУТЕЙ 42

Лабораторная работа № 9
ОПРЕДЕЛЕНИЕ КОНВЕЙЕРИЗИРУЕМЫХ ПУТЕЙ 46

Лабораторная работа № 10
ОПРЕДЕЛЕНИЕ МНОЖЕСТВ СВЕРТЫВАЕМЫХ ВЕРШИН ... 51

Лабораторная работа № 11
ОПРЕДЕЛЕНИЕ МНОЖЕСТВА АЛЬТЕРНАТИВНЫХ ВАРИАНТОВ ПРОЕКТИРУЕМОЙ СИСТЕМЫ 54

Лабораторная работа № 12
ПОСТРОЕНИЕ ВЫЧИСЛИТЕЛЬНОГО ГРАФА АЛГОРИТМА 61

Лабораторная работа № 13
ПЕРВАЯ ПРОВЕРКА РЕАЛИЗУЕМОСТИ ВЫЧИСЛИТЕЛЬНОЙ СИСТЕМЫ РЕАЛЬНОГО ВРЕМЕНИ 66

Лабораторная работа № 14
ФОРМИРОВАНИЕ ВЕКТОРА ВРЕМЕННОЙ РАЗВЕРТКИ 69

Лабораторная работа № 15
ВТОРАЯ ПРОВЕРКА РЕАЛИЗУЕМОСТИ ВЫЧИСЛИТЕЛЬНОЙ СИСТЕМЫ РЕАЛЬНОГО ВРЕМЕНИ 76

Лабораторная работа № 16
ПОСТРОЕНИЕ ВЫЧИСЛИТЕЛЬНОГО ГРАФА АЛГОРИТМА С РЕГИСТРОВЫМИ ФАЙЛАМИ 78

Лабораторная работа № 17
ФОРМИРОВАНИЕ ГРАФА ВЫЧИСЛИТЕЛЬНОЙ СТРУКТУРЫ 82

Лабораторная работа № 18
ФОРМИРОВАНИЕ УПРАВЛЯЮЩИХ ПАРАМЕТРОВ ВЫЧИСЛИТЕЛЬНОЙ СИСТЕМЫ 85

Лабораторная работа № 19
ВЫБОР ИЗ МНОЖЕСТВА АЛЬТЕРНАТИВНЫХ ВАРИАНТОВ, ОПТИМАЛЬНЫХ ПО ЗАДАННЫМ КРИТЕРИЯМ 88

ЛИТЕРАТУРА 94

Учебное издание

Кобайло Александр Серафимович

СИНТЕЗ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ РЕАЛЬНОГО ВРЕМЕНИ

ЛАБОРАТОРНЫЙ ПРАКТИКУМ

Учебно-методическое пособие

Редактор *О. П. Приходько*
Компьютерная верстка *О. П. Приходько*
Корректор *О. П. Приходько*

Подписано в печать 08.06.2012. Формат $60\times84^{1}/_{16}$.
Бумага офсетная. Гарнитура Таймс. Печать офсетная.
Усл. печ. л. 5,6. Уч.-изд. л. 5,8.
Тираж 100 экз. Заказ .

Издатель и полиграфическое исполнение:
УО «Белорусский государственный технологический университет».
ЛИ № 02330/0549423 от 08.04.2009.
ЛП № 02330/0150477 от 16.01.2009.
Ул. Свердлова, 13а, 220006, г. Минск.